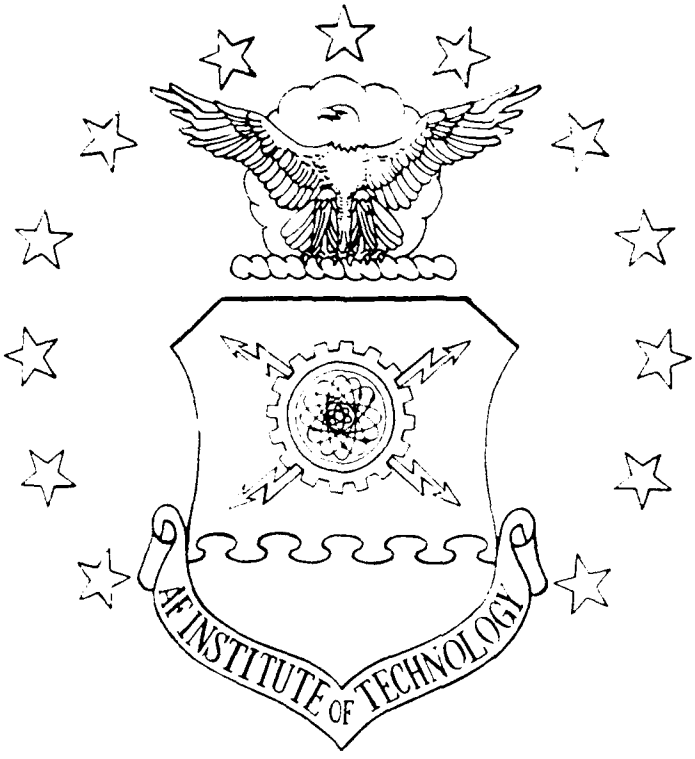

NUMERICAL ANALYSIS OF AN AXISYMMETRIC THRUST AUGMENTING EJECTOR

THESIS

Kenneth R. Gage, Captain, USAF

AFIT/GAE/ENY/89D-10

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/89D-10

NUMERICAL ANALYSIS OF AN AXISYMMETRIC THRUST AUGMENTING EJECTOR

THESIS

Kenneth R. Gage, Captain, USAF
AFIT/GAE/ENY/89D-10

AFIT/GAE/ENY/89D-10

NUMERICAL ANALYSIS OF AN AXISYMMETRIC THRUST AUGMENTING EJECTOR

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Aeronautical Engineering

Kenneth R. Gage, B.S.
Captain, USAF

December 1990

Acknowledgments

This project has been a long time in work and has been full of ups and downs, more than I care to recall. Through it all, the support of a few people have kept it, and me, all going. I would like to take this opportunity to thank them.

To my advisor, Dr. Franke, for supporting me in taking on a project which interested me although it was somewhat out of his field.

To Dr. Joe Shang, for giving me access to the resources and personnel to help me along and for the guiding hand when I needed it most.

To Dr. Don Rizzetta, for helping me iron out the little technicalities.

Especially to Dr. Kervyn Mach, for setting up and guiding me through the myriad complexities of working on an unfamiliar system and for helping me to understand the program I had to modify!

But mostly to my wife Barbara, who had to put up with what seemed like an interminable wait.

Table of Contents

Table of Contents (cont.)

List of Figures

List of Figures (cont.)

List of Tables

Notation

A	Cross-Sectional Area of Mixing Region
A_E	Cross-Sectional Area of (Diffuser) Exit
A_j	Cross-Sectional Area of Jet
A^*	Critical Area
c_v	Specific Heat of Constant Volume
D	Van Driest Damping Factor
e	Energy
F	Thrust Force
$F_{noz\ isen}$	Isentropic Thrust of Primary Nozzle
F_{wake}	Wake Function
j	Radial Grid Index
k	Axial Grid Index
L	Scaling Length
L_1, L_2	Diffuser Section Lengths
L_i	Nozzle Position
M, N	Integers
P	Static Pressure
$P_{o\ noz}$	Nozzle Stagnation Pressure
P_1	Static Pressure at Jet Exit Plane
P_a	Ambient Pressure
P_{rat}	Pressure Ratio
Pr	Prandtl Number
Pr_t	Turbulent Prandtl Number
R	Gas Constant for Air
r	Radial Cylindrical Coordinate
T	Static Temperature
T_a	Ambient Temperature
T_w	Constant Wall Temperature
t	Time
u	Axial Component of Velocity
U_{ref}	Reference Velocity

Notation (cont.)

V	Total Velocity
V_1	Secondary Flow Velocity at Jet Exit Plane
V_E	Velocity at Exit
V_J	Jet Velocity
v	Radial Component of Velocity
x	Axial Cylindrical Coordinate
y^*	Critical Distance from Wall
α	Nozzle Area Ratio A_J / A
α_1	Flow Injection Angle
β	Diffuser Area Ratio A_E / A
γ	Ratio of Specific Heats
ϵ	Turbulent Viscosity
ζ	'Axial' Curvilinear Coordinate
η	'Radial' Curvilinear Coordinate
θ	Nozzle Position Angle
μ	Viscosity
ρ	Density
σ	Normal Stresses
τ	Shear Stresses
ϕ	Thrust Augmentation Ratio
ψ_1, ψ_2	Diffuser Angles
ω	Vorticity

Abstract

Use of an ejector is an effective way to increase the thrust produced by a jet. In this thesis project an axisymmetric ejector concept which had been previously explored by experiment was numerically modeled. An existing axisymmetric, internal flow code based on the explicit MacCormack method was modified to incorporate primary nozzle structure and flow injection within the flowfield. Results were compared qualitatively and quantitatively with experimental results to verify the validity of the model. Internal flow structure, difficult to obtain in experiment, is easily examined. This code may be used for parametric analysis of such ejector performance parameters as primary nozzle location, flow injection angle, diffuser area ratio, and inlet geometry to optimize future hardware configurations.

NUMERICAL ANALYSIS OF AN AXISYMMETRIC THRUST AUGMENTING EJECTOR

I Introduction

Background

Ejectors provide an attractive solution to a problem which has plagued designers of jet powered Vertical/Short Takeoff and Landing (V/STOL) aircraft since they were conceived. The problem being, that in order to get thrust greater than the weight of the vehicle such that vertical takeoff or landing can be achieved, an aircraft must be equipped with a powerplant which is larger (sometimes substantially) than that which is needed for its primary mission. Ejectors, due to their thrust augmenting characteristics, provide a means for achieving thrust to weight ratios greater than unity with a more reasonably sized powerplant.

In an ejector, a jet flow is directed into an open duct of somewhat larger cross-sectional area. A secondary flow is induced through the duct by viscous entrainment (Figure 1). A simple analysis developed by Von Karman (18:461-62) shows that the thrust theoretically achievable by such a process is up to twice that of the primary nozzle alone (Figure 2).

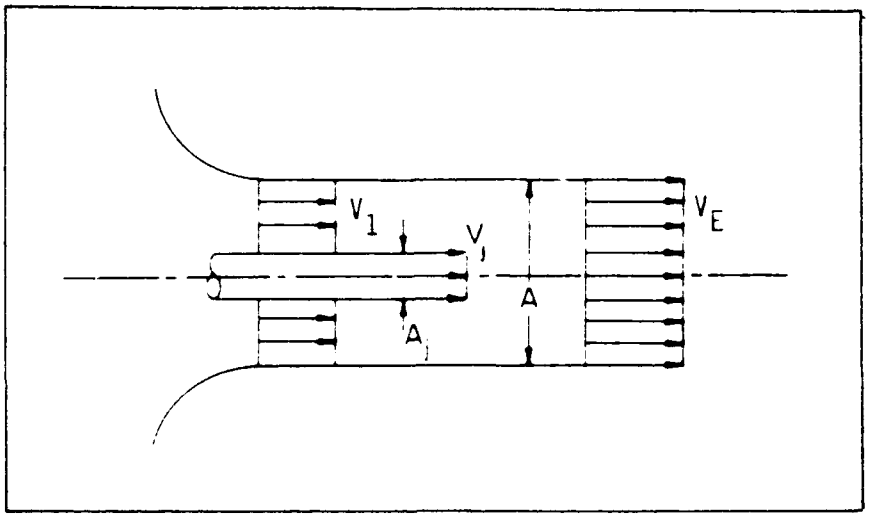

Fig. 1. Ejector Concept (9:281)

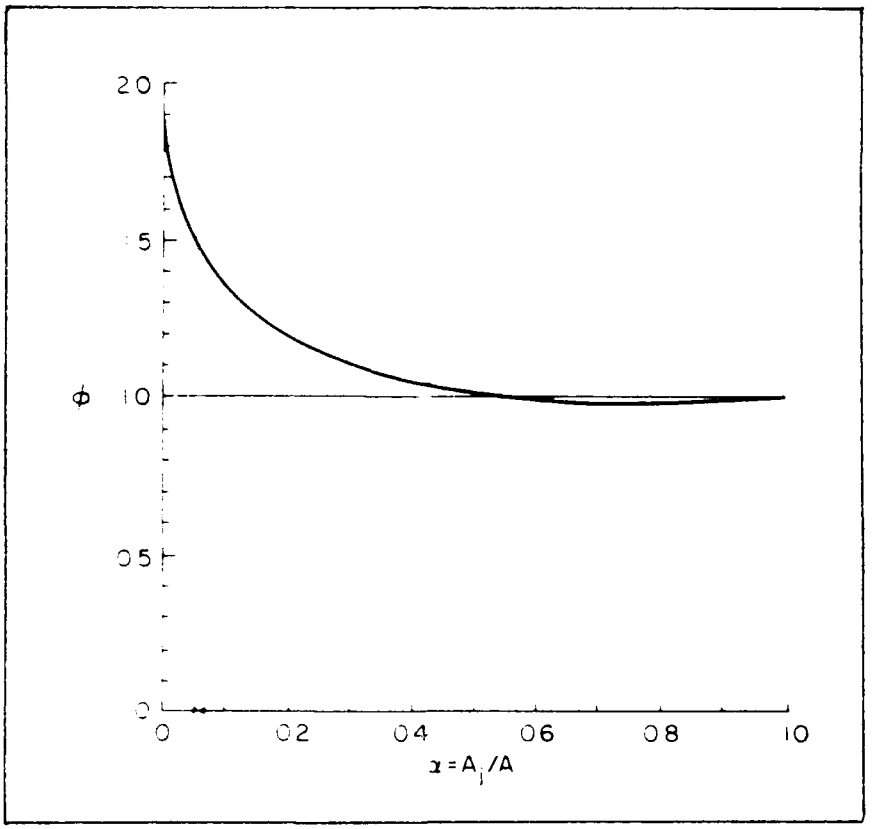

Fig. 2. Ideal Thrust Augmentation vs. α (9:281)

The thrust augmentation ratio (ϕ) is defined as the ratio of the total thrust of the ejector (F) to the isentropic thrust of the primary nozzle alone (assuming incompressible flow):

$$\phi = F / F_{noz\,isen}$$

$$\phi = \varrho\, V_E^2\, A \,/\, \varrho\, V_j^2\, A_j$$

$$\phi = (V_E/V_j)^2 \,/\, \alpha \qquad (1)$$

where α is the ratio of the area of the jet (A_J) to the area of the mixing chamber (A)

$$\alpha = A_j/A$$

and V_E/V_J is the ratio of the flow velocity following complete mixing (V_E) to the flow velocity of the primary nozzle (V_J) found by applying continuity from the jet exit plane to the exit of the ejector:

$$V_1\,(A - A_j) + V_j\,A_j = V_E\,A \qquad (2)$$

and applying conservation of momentum:

$$A\,(P_1 - P_0) = \varrho\,A\,V_E^2 - \varrho\,A_j\,V_j^2 - \varrho\,(A - A_j)\,V_1^2 \qquad (3)$$

Applying Bernoulli's equation from ahead of the inlet to the jet exit plane:

$$P_0 = P_1 + \tfrac{1}{2}\,\varrho\,V_1^2 \qquad (4)$$

gives three equations in three unknowns, V_E, V_1, and P_1.

Eliminating V_1 and P_1, the velocity ratio, V_E/V_J, is found to be:

$$V_E/V_j = \frac{[\,-\alpha\,(\,1-2\alpha\,)+(\,2\alpha-6\alpha^2+6\alpha^3-2\alpha^4\,)^{1/2}\,]}{(\,1-2\alpha+2\alpha^2\,)} \qquad (5)$$

(7:28)

The thrust augmentation achieved may be further increased through the use of a diffuser (Figure 3).

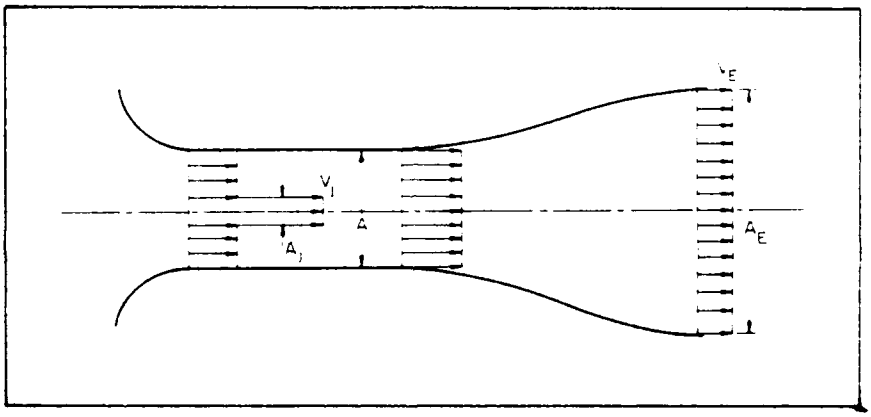

Fig. 3. Ejector with Diffuser (9:283)

Through a similar analysis, the modified equation for a diffuser equipped ejector is found to be:

$$(V_E/V_j)^2 + (V_E/V_j)\left[\frac{2\alpha\beta\,(1-2\alpha)}{1-2\alpha+\alpha^2(1+\beta^2)}\right] - \left[\frac{2\alpha-3\alpha^2}{1-2\alpha+\alpha^2(1+\beta^2)}\right] = 0 \qquad (6)$$

(9:282)

with ϕ now given by (assuming incompressible flow):

$$\phi = \varrho V_E^2 A_E / \varrho V_j^2 A_j$$

$$\phi = (V_E/V_j)^2 (A_E/A) (A/A_j)$$

$$\phi = (V_E/V_j)^2 (\beta/\alpha) \qquad (7)$$

where:

$$\beta = A_E/A$$

with V_E and A_E as defined in Figure 3 (9:282-3).

Plotting the thrust augmentation ratio shows that for every primary nozzle to mixing chamber area ratio (A_J/A) there is an optimum diffuser area ratio (A_E/A) which provides the maximum thrust augmentation (Figure 4). Within the practical limit of area ratios, it can be seen that thrust augmentations of up to 3 times the primary jet thrust are theoretically achievable.

These theoretical values, however, assume that the ejector is long enough that complete mixing of the primary and secondary flows takes place. In practice, the size of the ejector is limited by aircraft installation constraints and frictional effects, which lead to separations in long diffusers. In such short ejectors mixing is incomplete and coanda nozzles, in which the jet flow is injected along the wall (Figure 5), have proven to be more effective due to the greater mixing surface area of the primary jet. (9:286)

Coanda nozzles provide an additional benefit for ejectors equipped with a diffuser in that the high energy flow along the wall helps to prevent separation in the diffuser allowing greater area ratios to be achieved.

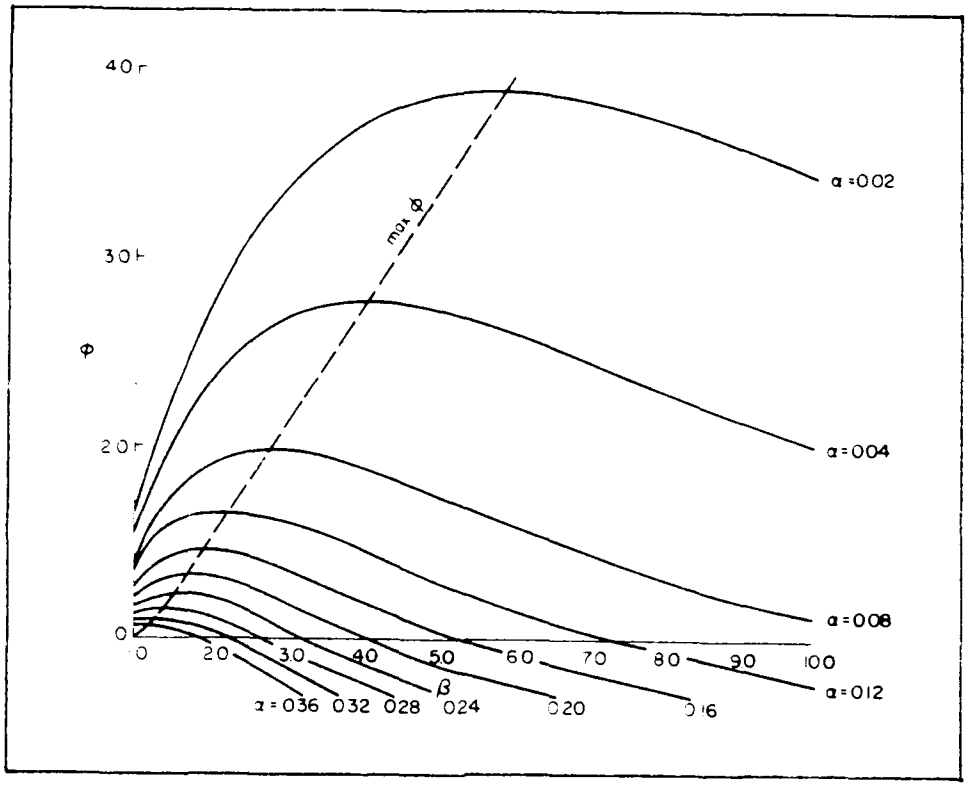

Fig. 4. Thrust Augmentation as a Function of α and β. (9:284)

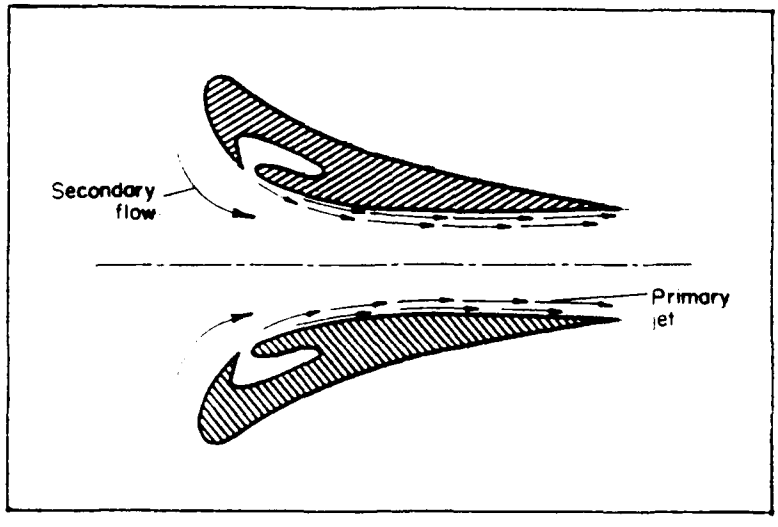

Fig. 5. Ejector with Coanda Nozzles (9:285)

In a previous study, Kedem, (6) experimentally investigated a two dimensional ejector with center and coanda nozzles and found that coanda nozzles did provide better performance (6:51). Reznick (11) and later Unnever (17) expanded on these results by testing both two dimensional and circular ejectors with coanda nozzles. The circular ejectors provided the best performance, achieving a thrust augmentation ratio of 2.09. This was accomplished through the use of discrete jets located about the periphery of the inlet which are referred to as 'hook nozzles' because of their hook-like shape (11:44). This approach was further investigated by Lewis (8) and later Uhuad (16). Hardware limitations necessarily constrained the various configurations which could be tested. It is desirable to have a computational model of a circular ejector which is free of such limitations. With this model a full range of inlet and diffuser geometries, nozzle sizes and positions, and nozzle plenum pressures can be investigated. The computational solutions also provide complete data on the behavior of the flow within the ejector, which is difficult, if not impossible, to obtain experimentally. This model has, however, been constrained to an axisymmetric case to reduce complexity and required computational power. This eliminates the possibility of modeling the hook nozzles used to achieve the high thrust augmentation observed by Reznick. This model is therefore constrained to the case of an annular nozzle, which was also one of the configurations tested by both Reznick and Unnever.

Purpose and Objectives

The primary objectives of this study are:

1. Develop a numerical algorithm that models a coanda flow axisymmetric ejector of the type tested by Reznick and Unnever.

2. Validate the model by demonstrating agreement with experimental results.

3. Provide data visualization of internal flow characteristics.

4. Provide sufficient flexibility to accomplish secondary objectives.

5. Begin preliminary investigation of secondary objectives.

The secondary objectives are:

1. Investigate the geometrical and fluid injection parameters which effect ejector performance. The parameters which may be investigated include:

   a) Fluid injection angle
   b) Nozzle position
   c) Nozzle area ratio
   d) Diffuser area ratio
   e) Diffuser geometry
   f) Pressure ratio ($P_{o\ noz}$ / P_a)

2. From this parametric analysis, define new ejector configurations which may merit further experimental investigation.

Scope

A generic axisymmetric, internal, Navier-Stokes flow solver based on the explicit MacCormack method provided by WRDC/FIMM was modified to incorporate primary nozzle structure and flow injection within the flowfield (program JSIAXK, App B). A diffuser geometry, nozzle pressure ratio, and nozzle exit area and location were chosen for each run. The program was then run to a steady-state solution for a given flow injection angle (α_1). The injection angle was then varied and the run continued to new steady-state solutions to collect data on α_1 - variation of augmentation. Additional runs were made with different nozzle locations holding pressure ratio, nozzle area, and nozzle distance from the wall constant to provide data on nozzle location effects on augmentation. Also, runs were made with different nozzle areas for the same location to provide data on area ratio effects. Runs with different geometries and pressure ratios could provide data on variations with these parameters.

The achieved thrust was calculated by a control volume analysis (Figure 6). This analysis results in the equation:

$$F = \int_A \rho u^2 dA \qquad (8)$$

where the right side of the equation is the integral of the x - momentum at the exit (5:12-13). The achieved thrust was compared to the isentropic thrust of the primary nozzle alone to calculate the thrust augmentation ratio as in equation (5). Thrust augmentation ratios were compared directly with experimental data for similar configurations and the differences analyzed. Exit plane velocity distributions and variation of thrust augmentation with α_1

and position were compared qualitatively with available experimental results.

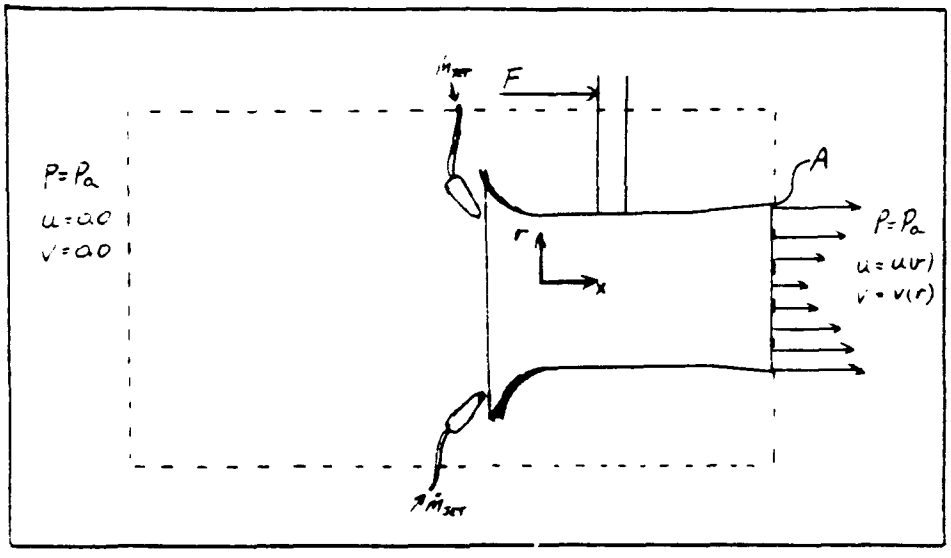

Fig. 6. Control Volume

II COMPUTATIONAL APPROACH AND PROCEDURES

Ejector Geometry

The geometry of the axisymmetric ejector which was investigated by Reznick and Unnever and was modeled in this study is portrayed in Figure 7. The mixing chamber was 4.4 inches in diameter and 3 inches in length, the inlet lip had a 2-inch radius of curvature, and the multi-section diffuser could be changed to alter the diffuser angles ($\Psi_1, \Psi_2, \ldots$) and lengths ($L_1, L_2, \ldots$), up to three sections, and thus the area ratio A_E/A. The annular exit of the nozzle had a gap of 0.065 inches and a radius of 3.0025 inches and could be positioned at any distance (L_i) from the ejector throat (constant area mixing chamber). (17:52)

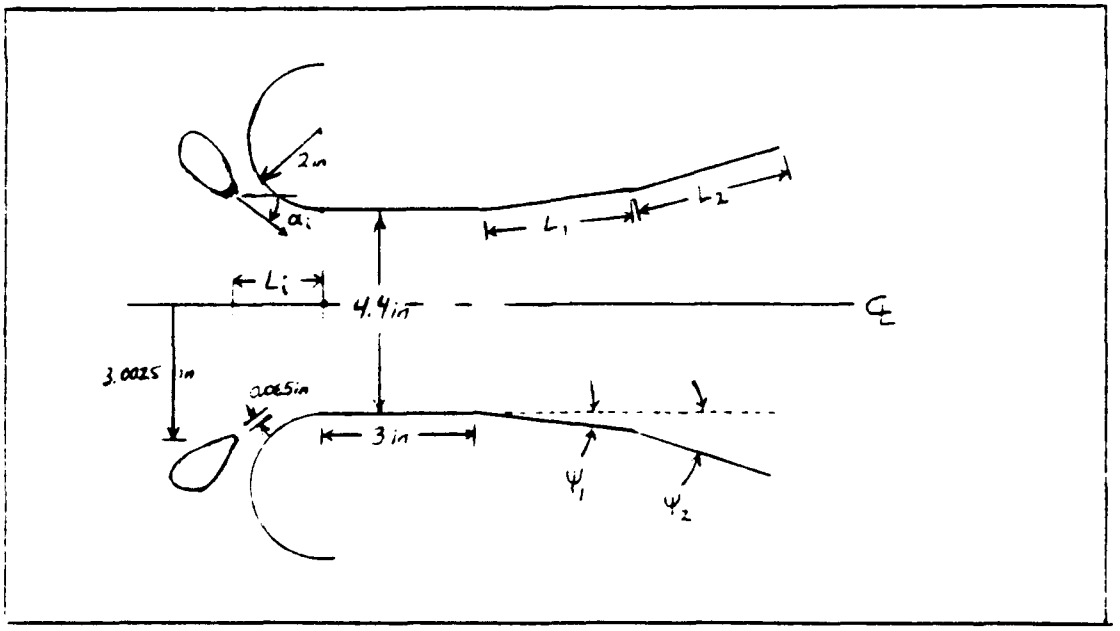

Fig. 7. Ejector Geometry

Grid

The computational grid was generated using the program EJECGRD (Appendix A-1) provided by WRDC/FIMM and modified for the required geometry. Taking advantage of axial symmetry, the grid models a single radial/axial plane from the centerline to the wall (Figure 8).

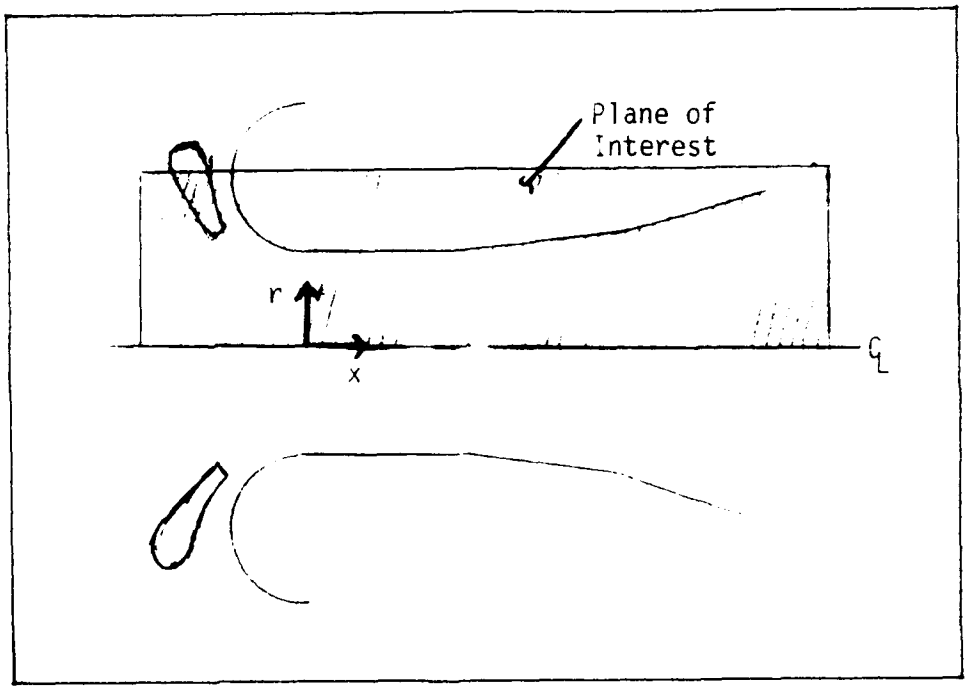

Fig. 8. Plane of Modeled Flow

The vertical (radial) grid lines are defined as follows:

For the inlet, by circular arcs perpendicular to the wall and centerline.

For the mixing area and diffuser, by parabolas perpendicular to the wall and centerline (which degenerate to straight lines for the mixing area).

Spacing between horizontal grid lines was varied exponentially to refine the grid near the wall where large gradients are found due to the jet. The grid was also refined near the centerline to minimize the effects of the boundary conditions. Spacing between vertical grid lines was varied exponentially to refine the grid in the vicinity of the inlet (and the jet exit) to help offset the tendency of the grid to spread out due to the increasing radii of the circular arcs in the inlet and also to concentrate on the larger gradients upstream. The exponential spacing was accomplished using the program DEHNEN (Appendix A-2) also provided by WRDC/FIMM which was used to produce vertical and horizontal grid spacing tables, EJECVRT and EJECHRZ (Appendices A-3,-4). Once a nozzle location was identified, the grid was refined both vertically and horizontally in the area of the nozzle exit. An example of the grid is shown in Figure 9. Note that only every fourth grid line has been drawn for the purpose of clarity.

Flowfield Initialization

The program EJECGRD also calculates an initial flowfield based on a simple one-dimensional flow area ratio formula given a value of the critical area (A*), the duct area through which the flow would theoretically be sonic. This critical area was chosen to provide a reference Mach number at the diffuser exit of 0.1 (111.7 ft/sec). This program was modified to overlay a high-speed jet flow over the low speed duct flow calculated by this procedure. This

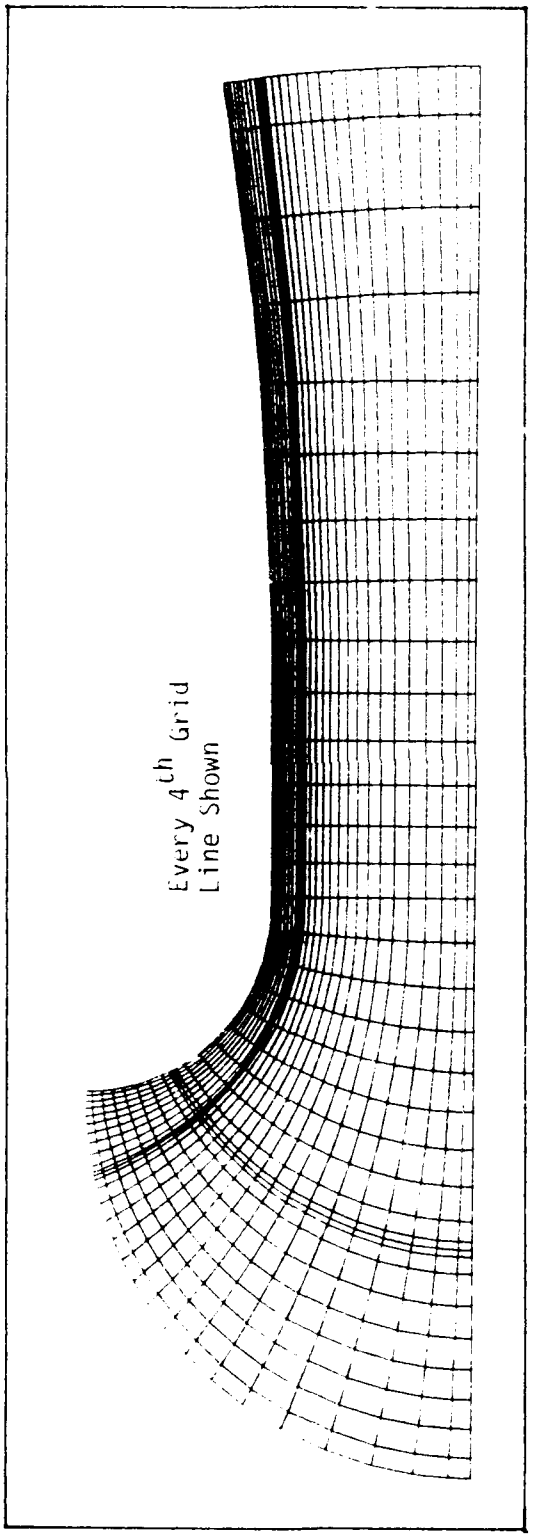

Fig. 9. Computational Grid

was necessary because the formula used assumes a mass flow balance between the beginning and end of the duct. If the jet flow is not accounted for, this provides an extremely poor initial flowfield which can lead to extended processing time or even cause the solution to 'blow up', or diverge, as the flowfield tries to adapt to the increased mass flow injected by the primary nozzle.

Flow Solver

The flow solving program JSIAXK (Appendix B) solves the conservative form of the Navier-Stokes equations using the explicit MacCormack scheme. The conservative Navier-Stokes equations in axisymmetric coordinates are (3:3):

Equation of Mass

$$\frac{\partial}{\partial t}(\varrho) + \frac{\partial}{\partial x}(\varrho u) + (1/r)\frac{\partial}{\partial r}(\varrho v r) = 0 \qquad (9)$$

Equation of Axial Motion

$$\frac{\partial}{\partial t}(\varrho u) + \frac{\partial}{\partial x}(\varrho u^2) + (1/r)\frac{\partial}{\partial r}(\varrho u v r) = \frac{\partial}{\partial x}(\sigma_{xx}) + (1/r)\frac{\partial}{\partial r}(r\tau_{xr}) \qquad (10)$$

Equation of Radial Motion

$$\frac{\partial}{\partial t}(\varrho v) + \frac{\partial}{\partial x}(\varrho u v) + (1/r)\frac{\partial}{\partial r}(\varrho v^2 r) = \frac{\partial}{\partial x}(\tau_{xr}) + (1/r)\frac{\partial}{\partial r}(r\sigma_{rr}) - \sigma_H \qquad (11)$$

Energy Equation

$$\frac{\partial}{\partial t}(\varrho e) + \frac{\partial}{\partial x}(\varrho u e) + (1/r)\frac{\partial}{\partial r}(\varrho v e r) = \frac{\partial}{\partial x}\left[\gamma \frac{\partial e}{\partial x}\left(\frac{\mu}{\mathrm{Pr}} + \frac{\epsilon}{\mathrm{Pr}_t}\right) - (u\sigma_{xx} + v\tau_{xr})\right]$$

$$+ (1/r)\frac{\partial}{\partial r}\left[r\gamma \frac{\partial e}{\partial r}\left(\frac{\mu}{\mathrm{Pr}} + \frac{\epsilon}{\mathrm{Pr}_t}\right) - r(u\tau_{xr} + v\sigma_{rr})\right] \quad (12)$$

These equations are written in vector form as (4:17):

$$\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}}{\partial x} + (1/r)\frac{\partial \vec{G}}{\partial r} = (1/r)\,\vec{H} \quad (13)$$

where:

$$\vec{U} = \begin{bmatrix} \varrho \\ \varrho u \\ \varrho v \\ \varrho e \end{bmatrix}$$

$$\vec{F} = \begin{bmatrix} \varrho u \\ \varrho u^2 - \sigma_{xx} \\ \varrho uv - \tau_{xr} \\ \varrho ue - \gamma \frac{\partial e}{\partial x}\left(\frac{\mu}{\mathrm{Pr}} + \frac{\epsilon}{\mathrm{Pr}_t}\right) - (u\sigma_{xx} + v\tau_{xr}) \end{bmatrix}$$

$$\vec{G} = \begin{bmatrix} \varrho vr \\ \varrho uvr - \tau_{xr} r \\ \varrho v^2 r - \sigma_{rr} r \\ \varrho ver - r\gamma \frac{\partial e}{\partial r}\left(\frac{\mu}{\mathrm{Pr}} + \frac{\epsilon}{\mathrm{Pr}_t}\right) - r(u\tau_{xr} + v\sigma_{rr}) \end{bmatrix}$$

$$\vec{H} = \begin{bmatrix} 0 \\ 0 \\ -\sigma_H \\ 0 \end{bmatrix}$$

The transformed vector equation for general curvilinear coordinates is:

$$\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial \vec{F}}{\partial \eta}\frac{\partial \eta}{\partial x} + (1/r)\left[\frac{\partial \vec{G}}{\partial \zeta}\frac{\partial \zeta}{\partial r} + \frac{\partial \vec{G}}{\partial \eta}\frac{\partial \eta}{\partial r}\right] = (1/r)\,\vec{H} \qquad (14)$$

Applying the explicit MacCormack scheme to the vector equation results in the following algorithm (4:27):

$$\vec{U}(\xi\ ,\ \eta\ ,\ t + \Delta t) = \mathbf{L}(\Delta t)\,\vec{U}(\xi\ ,\ \eta\ ,\ t) \qquad (15)$$

where L(Δt) is the two-dimensional numerical operator representing MacCormack's algorithm acting on the transformed conservation equations. Through use of time-splitting, this two-dimensional operator is separated into two one-dimensional sweep operators in the ζ - and η - directions. The operator $L_\zeta(\Delta t)$ represents the solution of the equation:

$$\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}}{\partial \xi}\frac{\partial \xi}{\partial x} + (1/r)\frac{\partial \vec{G}}{\partial \zeta}\frac{\partial \zeta}{\partial r} = 0 \qquad (16)$$

in the ζ - direction by a time increment of Δt seconds. In a like manner, the operator $L_\eta(\Delta t)$ represents the solution of:

$$\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}}{\partial \eta}\frac{\partial \eta}{\partial x} + (1/r)\frac{\partial \vec{G}}{\partial \eta}\frac{\partial \eta}{\partial r} = (1/r)\,\vec{H} \qquad (17)$$

in the η - direction by a time increment of Δt seconds.

The dependent variable vector U(ζ , η , t) can then be advanced in time as:

$$\vec{U}(\zeta\ ,\ \eta\ ,\ t + \Delta t) = [\ \mathbf{L}_{\zeta}^{M/2}(\Delta t/M)\ \mathbf{L}_{\eta}(\Delta t)\ \mathbf{L}_{\zeta}^{M/2}(\Delta t/M)]\ \vec{U}(\zeta\ ,\ \eta\ ,\ t) \qquad (18)$$

with $\Delta t = \Delta t_{\zeta}$ if $\Delta t_{\zeta} < \Delta t_{\eta}$

or as:

$$\vec{U}(\zeta\ ,\ \eta\ ,\ t + \Delta t) = [\ \mathbf{L}_{\eta}^{N/2}(\Delta t/N)\ \mathbf{L}_{\zeta}(\Delta t)\ \mathbf{L}_{\eta}^{N/2}(\Delta t/N)]\ \vec{U}(\zeta\ ,\ \eta\ ,\ t) \qquad (19)$$

with $\Delta t = \Delta t_{\eta}$ if $\Delta t_{\eta} < \Delta t_{\zeta}$

where M and N are the smallest even integers of the quotients ($\Delta t_{\eta}/\Delta t_{\zeta}$) and ($\Delta t_{\zeta}/\Delta t_{\eta}$) respectively. The timesteps Δt_{ζ} and Δt_{η} are the maximum allowable timesteps in the ζ - and η - directions as determined by stability requirements.

The finite difference form of these sweep operators consist of a two step predictor - corrector procedure which uses one-sided differencing in the direction of sweep and central differencing in the direction perpendicular to the direction of sweep. When complete, this method is equivalent to a second order central differencing scheme in two dimensions.

The $L_\zeta(\Delta t)$ sweep operator represents the following numerical procedure (4:29):

PREDICTOR

$$(\vec{U}_{k,j})^{n+\frac{1}{2}} = (\vec{U}_{k,j})^n - \frac{\Delta t}{\Delta \zeta}[(\vec{F}_{k,j})^n - (\vec{F}_{k-1,j})^n]\,(\frac{\partial \zeta}{\partial x})_{k,j}$$

$$- (1/r)\frac{\Delta t}{\Delta \zeta}[r_{k,j}\,(\vec{G}_{k,j})^n - r_{k,j}\,(\vec{G}_{k-1,j})^n]\,(\frac{\partial \zeta}{\partial r})_{k,j} \qquad (20)$$

CORRECTOR

$$(\vec{U}_{k,j})^{n+1} = \tfrac{1}{2}\{\,(\vec{U}_{k,j})^n + (\vec{U}_{k,j})^{n+\frac{1}{2}} - \frac{\Delta t}{\Delta \zeta}[(\vec{F}_{k+1,j})^{n+\frac{1}{2}} - (\vec{F}_{k,j})^{n+\frac{1}{2}}]\,(\frac{\partial \zeta}{\partial x})_{k,j}$$

$$- (1/r)\frac{\Delta t}{\Delta \zeta}[r_{k,j}\,(\vec{G}_{k+1,j})^{n+\frac{1}{2}} - r_{k,j}\,(\vec{G}_{k,j})^{n+\frac{1}{2}}]\,(\frac{\partial \zeta}{\partial r})_{k,j}\,\} \qquad (21)$$

Similarly, the $L_\eta(\Delta t)$ operator represents the following predictor - corrector steps (4:29,31):

PREDICTOR

$$(\vec{U}_{k,j})^{n+\frac{1}{2}} = (\vec{U}_{k,j})^n - \frac{\Delta t}{\Delta \eta}[(\vec{F}_{k,j})^n - (\vec{F}_{k,j-1})^n]\,(\frac{\partial \eta}{\partial x})_{k,j}$$

$$- (1/r)\frac{\Delta t}{\Delta \eta}[r_{k,j}\,(\vec{G}_{k,j})^n - r_{k,j}\,(\vec{G}_{k,j-1})^n]\,(\frac{\partial \eta}{\partial r})_{k,j} + (1/r)\,\Delta t\,(\vec{H}_{k,j})^n \qquad (22)$$

CORRECTOR

$$(\vec{U}_{k,j})^{n+1} = \tfrac{1}{2}\{(\vec{U}_{k,j})^{n} + (\vec{U}_{k,j})^{n+\frac{1}{2}} - \frac{\Delta t}{\Delta \eta}[(\vec{F}_{k,j+1})^{n+\frac{1}{2}} - (\vec{F}_{k,j})^{n+\frac{1}{2}}](\frac{\partial \eta}{\partial x})_{k,j}$$

$$- (1/r)\frac{\Delta t}{\Delta \eta}[r_{k,j}(\vec{G}_{k,j+1})^{n+\frac{1}{2}} - r_{k,j}(\vec{G}_{k,j})^{n+\frac{1}{2}}](\frac{\partial \eta}{\partial r})_{k,j} + (1/r)\,\Delta t\,(\vec{H}_{k,j})^{n}\} \qquad (23)$$

where ^{n} and $^{n+1}$ designate values at time t and t + Δt respectively and $^{n+1/2}$ indicates a temporary, predicted value. The values $\frac{\partial \zeta}{\partial x}$, $\frac{\partial \zeta}{\partial r}$, $\frac{\partial \eta}{\partial x}$, and $\frac{\partial \eta}{\partial r}$ depend only on the grid geometry and are calculated only once for each grid point.

Variants of this program for two-dimensional internal flow and three-dimensional external flow have been used in studies by Olson, McGowan, and MacCormack (10); Shang, Hankey and Smith (13); and Shang and MacCormack (14). The internal axisymmetric version of the code was developed for in-office use by WRDC/FIMM.

Incorporation of Nozzle Structure

In order to model the ejector geometry desired, it was necessary to identify boundary conditions which accounted for the nozzle structure and the flow injection at the nozzle exit in addition to the usual wall and centerline conditions. Once the desired location of the nozzle exit was selected, the grid was refined in that area to better handle the large gradients which exist at the nozzle exit. Then, the grid points which best approximated the corners of the desired nozzle exit were identified (Figure 10).

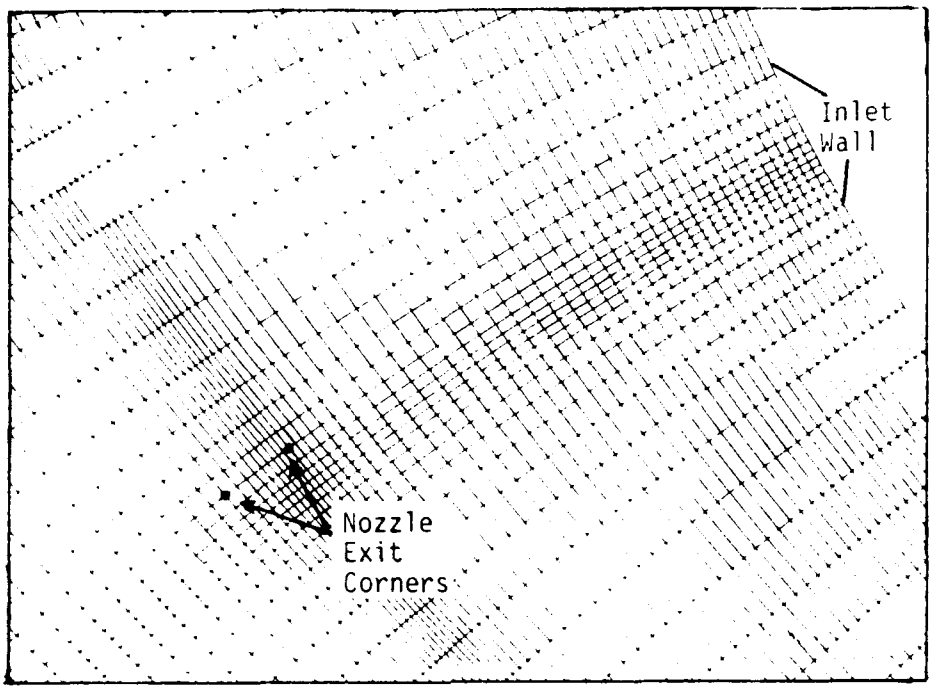

Fig. 10. Identification of Nozzle Exit Corners

The outside walls of the nozzle structure were defined as being three grid lines outside the nozzle exit. The outside walls and ends of the nozzle structure were defined as SOLID WALLS for the boundary conditions (Figure 11).

The grid points in the gap were defined as the NOZZLE EXIT for the boundary conditions. The area enclosed within the nozzle structure did not effect the computation of the flowfield and was set to zero velocity. Figure 12 shows a diagram of the grid boundaries with the types of boundary conditions applied identified.

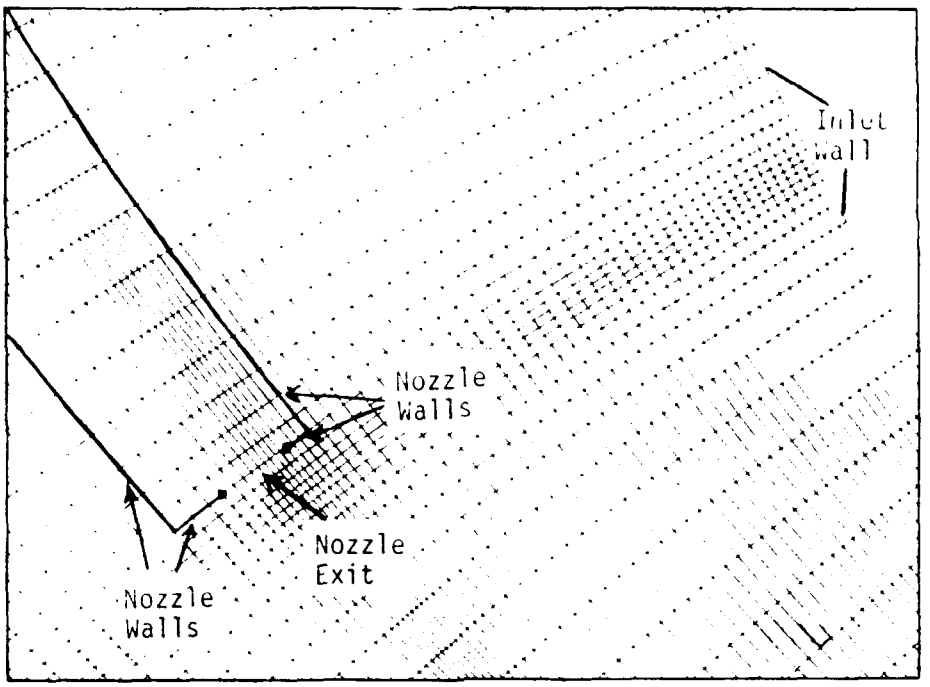

Fig. 11. Identification of Nozzle Walls

Boundary Conditions

Boundary conditions were imposed to accommodate the geometrical restrictions on the flowfield by defining the solid surfaces, to incorporate primary nozzle flow based on a constant plenum pressure, and to provide the necessary upstream and downstream conditions to achieve closure.

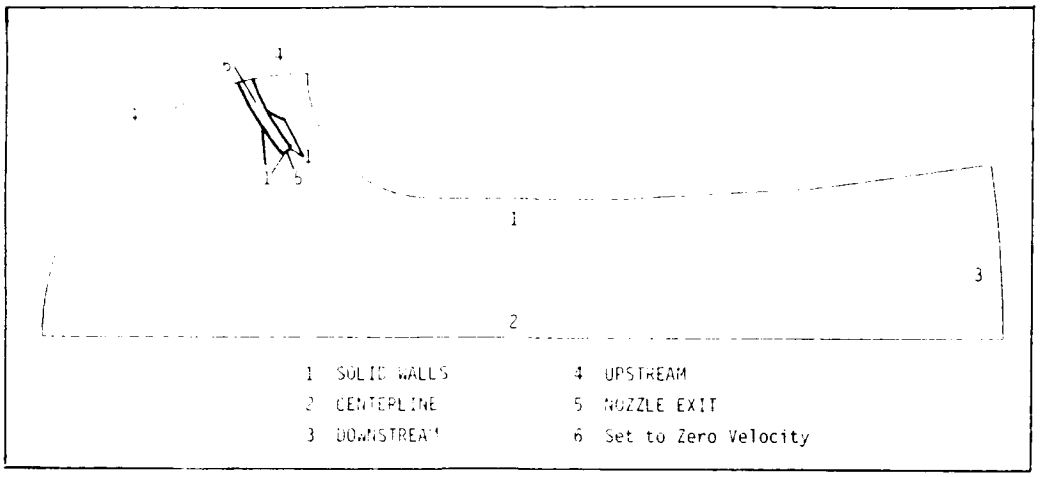

Fig. 12. Application of Boundary Conditions

Solid Walls. For the wall of the ejector and the inner and outer walls of the primary nozzle, applying the conditions of no penetration and no slip provides:

$$(\varrho u)_w = 0 \tag{24}$$

$$(\varrho v)_w = 0 \tag{25}$$

Allowing no normal pressure gradient ($\frac{\partial P}{\partial \eta} = 0$) leads to:

$$(\varrho e)_w = \varrho_1 \left[e_1 - \tfrac{1}{2} (u_1^2 + v_1^2) \right] \tag{26}$$

Applying a constant wall temperature T_w:

$$(\varrho)_w = (\varrho e)_w / (c_v T_w) \tag{27}$$

where the subscript w indicates the value at the wall and the subscript 1 indicates a value taken one grid point away from the wall.

Centerline. Since the centerline is the axis of symmetry, there can be no cross velocity. This results in the equation:

$$(\varrho v)_{CL} = 0 \tag{28}$$

Also, there can be no radial gradients of u, ϱ, or T ($\frac{\partial u}{\partial \eta} = \frac{\partial \varrho}{\partial \eta} = \frac{\partial T}{\partial \eta} = 0$), resulting in the following equations:

$$\varrho_{CL} = \varrho_1 \tag{29}$$

$$(\varrho u)_{CL} = (\varrho u)_1 \tag{30}$$

$$(\varrho e)_{CL} = (\varrho e)_1 - \tfrac{1}{2} (\varrho v)_1^2 / \varrho_{CL} \tag{31}$$

where the subscript CL indicates the value at the centerline and the subscript 1 indicates a value taken one grid point away from the centerline.

Downstream. Enforcing conservation of mass in the ζ - direction ($\frac{\partial \varrho u}{\partial \zeta} + 1/r \frac{\partial \varrho v r}{\partial \zeta} = 0$) by allowing $\frac{\partial \varrho u}{\partial \zeta} = 0$ and $\frac{\partial \varrho v r}{\partial \zeta} = 0$ gives:

$$(\varrho u)_K = (\varrho u)_{K-1} \tag{32}$$

$$(\varrho v)_K = (\varrho v r)_{K-1} / r_K \tag{33}$$

where the subscript K indicates the value at the downstream boundary and the subscript K-1 indicates a value taken one grid point upstream. Since the flow in the ejector modeled is exclusively subsonic, the exit static pressure must equal

the ambient pressure P_a. Applying this constraint and allowing temperature to propagate downstream leads to:

$$(\varrho)_K = P_a / RT_{K-1} \tag{34}$$

$$(\varrho e)_K = \varrho_K [(c_v T_{K-1}) + \tfrac{1}{2} (u_K^2 + v_K^2)] \tag{35}$$

Upstream. Enforcing conservation of mass in the ζ - direction as was done on the downstream boundary leads to:

$$(\varrho u)_1 = (\varrho u)_2 \tag{36}$$

$$(\varrho v)_1 = (\varrho v r)_2 / r_1 \tag{37}$$

where the subscript 1 indicates the value at the upstream boundary and the subscript 2 indicates a value taken one grid point downstream. Additionally, assuming incompressible, isentropic flow from ambient stagnation conditions results in the equations:

$$(\varrho)_1 = P_a / RT_a \tag{38}$$

$$(\varrho e)_1 = \varrho_1 [(c_v T_a) + \tfrac{1}{2} (u_1^2 + v_1^2) (\gamma - 1)/\gamma] \tag{39}$$

where P_a and T_a are the ambient stagnation pressure and temperature respectively.

Nozzle Exit. The nozzle exit was treated the same as the upstream boundary except that the nozzle total pressure ($P_{0\ noz}$) was based on the defined pressure ratio (P_{rat}):

$$P_{0\,noz} = P_{rat}(P_a) \tag{40}$$

and the direction of the flow was constrained to the desired injection angle (α_1).

Turbulence model

The turbulence model used in the original program was a modified Baldwin-Lomax model based on the upper surface. This turbulence model uses a two region calculation of the eddy viscosity (14:4). The inner region calculation ϵ_i is used up to the distance y* from the wall after which the outer region formulation ϵ_o is used. The distance y* is the defined as the smallest distance from the wall at which ϵ_i and ϵ_o are equal.

The inner formulation is given by (14:4):

$$\epsilon_i = \varrho\,(0.4\,L\,D)^2\,|\omega| \tag{41}$$

where the vorticity of the flow, ω, the Van Driest damping factor, D, and the scaling length, L, are given by:

$$\omega = \tfrac{1}{2}\,\nabla \times \vec{U} \tag{42}$$

$$D = 1 - \exp[\,(-\varrho_w\,|\omega_w|\,/\,\mu_w)^{1/2}\,L\,/\,26\,] \tag{43}$$

$$L = [\,(x - x_w)^2 + (r - r_w)^2\,]^{1/2} \tag{44}$$

and the subscript w represents the value at the wall. The outer formulation is given by (14:4):

$$\epsilon_o = 0.0336\,\varrho\,F_{wake}\,/\,[\,1 + 5.5\,(0.3\,L/L_{max})^6\,] \tag{45}$$

where the wake function, F_{wake}, is the minimum value of the following at any point in space:

$$F_{wake} = L_{max}\,F_{max}$$

or

$$F_{wake} = 0.25\,L_{max}\,V_{max}^2\,/\,F_{max} \tag{46}$$

and L_{max} is the value of the length scale, L, where F:

$$F = L\,D\,|\omega| \qquad (47)$$

reaches its maximum value, F_{max}, within the turbulent shear layer.

To simplify this investigation, no attempt was made to identify better turbulence models for this type of wall jet flow. The turbulence model already incorporated in JSIAXK was not changed but was based on the nearest wall to include nozzle wall effects.

Procedure

Once the desired location of the nozzle exit was identified, the vertical and horizontal grid spacing tables, EJECVRT and EJECHRZ (Appendices A-3,A-4), were modified to refine the grid in that vicinity. EJECGRD was then run to produce the grid and establish the initial flowfield. Setting the flow injection angle (α_1) to approximate the flow direction established by the simplified initial flowfield (to minimize extreme gradients) the flow solver JSIAXK could be started.

Initially, the timestep per iteration was kept small until the larger gradients could be smoothed out, then was increased to provide faster convergence to the steady state. Once the large gradients of the initial flowfield were smoothed out, it was also possible to change α_1 until the desired value was set. The program was then run until steady state was reached. Steady state was defined for the purpose of this investigation as a change of less than 0.001 ft/sec in total velocity at each of four critical locations in 3000 iterations. This process took approximately 90,000

iterations to complete at 0.151 CPU seconds per iteration on the CRAY XMP computer. This solution was then stored on the Central Datafile System. A new α_1 was then set to the next angle of interest. Convergence on a new steady state solution then took approximately 3600 iterations.

This process was continued until solutions were achieved for all desired injection angles for the configuration under investigation. For a new nozzle location, the process was restarted from the beginning. Configurations 1a, 1b, and 1c were able to be run consecutively because the changes to the flowfield were minimal in switching from one to the other.

The post-processing program AUGMENT was then run for each flowfield solution. This program integrates the x - momentum across the exit and compares it to the isentropic thrust of the nozzle alone to calculate the thrust augmentation per the control volume analysis described in section I. A sample comparison of mass flows in and out was also accomplished to ensure that conservation of mass was being satisfied.

III. Results and Discussion

Cases Investigated

The cases investigated consist of four nozzle location configurations fc one ejector geometry. The geometry of the ejector was ,elected to be similar to one investigated by Unnever to allow for direct comparison. The dimensions of the ejector studied by Unnever (Figure 7) were (17:60):

$L_1 = 3.5$ in $\qquad \psi_1 = 3^\circ$

$L_2 = 3.5$ in $\qquad \psi_2 = 8^\circ$

resulting in a diffuser area ratio (β) of 1.71.

The dimensions of the annular nozzle of Unnever's experiment and those of the configurations investigated are tabulated in Table I. The dimensions of Unnever's nozzle indicate the it was 0.55 inches from the wall. Unnever's testing was carried out with a pressure ratio of 1.14 which was replicated for all configurations investigated.

Configuration 1a. The location and geometry of the nozzle for configuration 1 was chosen to approximate as closely as possible that of the annular nozzle tested by Unnever.

Configurations 0, 2, 3. The locations and geometries of the nozzles for configurations 0, 2, and 3 were chosen such that the distance from the wall and the nozzle area would be approximately the same as configuration 1a. This allowed investigation of the thrust augmentation effects of nozzle position angle, θ, only (Figure 13).

TABLE I

Nozzle Geometries

	L_1	Radius	Gap	A_j	θ	α
Unnever	2.25	3.00	0.065	1.226	61.9	0.081
Config 1a	2.25	2.96	0.072	1.349	61.9	0.089
Config 0	2.40	3.30	0.064	1.325	70.0	0.087
Config 2	1.96	2.51	0.081	1.296	50.0	0.085
Config 3	1.64	2.17	0.094	1.306	40.0	0.086
Config 1b	2.26	2.96	0.053	0.989	61.9	0.065
Config 1c	2.25	2.95	0.095	1.774	61.9	0.117

Configurations 1b, 1c. The locations of the nozzles for configurations 1b and 1c were chosen to be as close to that of configuration 1a as possible. The only change made was to the size of the gap and therefore the nozzle area. This was done to investigate nozzle area ratio effects on thrust augmentation.

Primary Objectives

Comparison with Experimental Data. Configuration 1a was designed to model as closely as possible the annular nozzle configuration used in Unnever's experimental investigation. A qualitative comparison of the exit velocity profile of configuration 1a with an experimentally measured profile from Reznick (Figs. 14 and 15) shows that

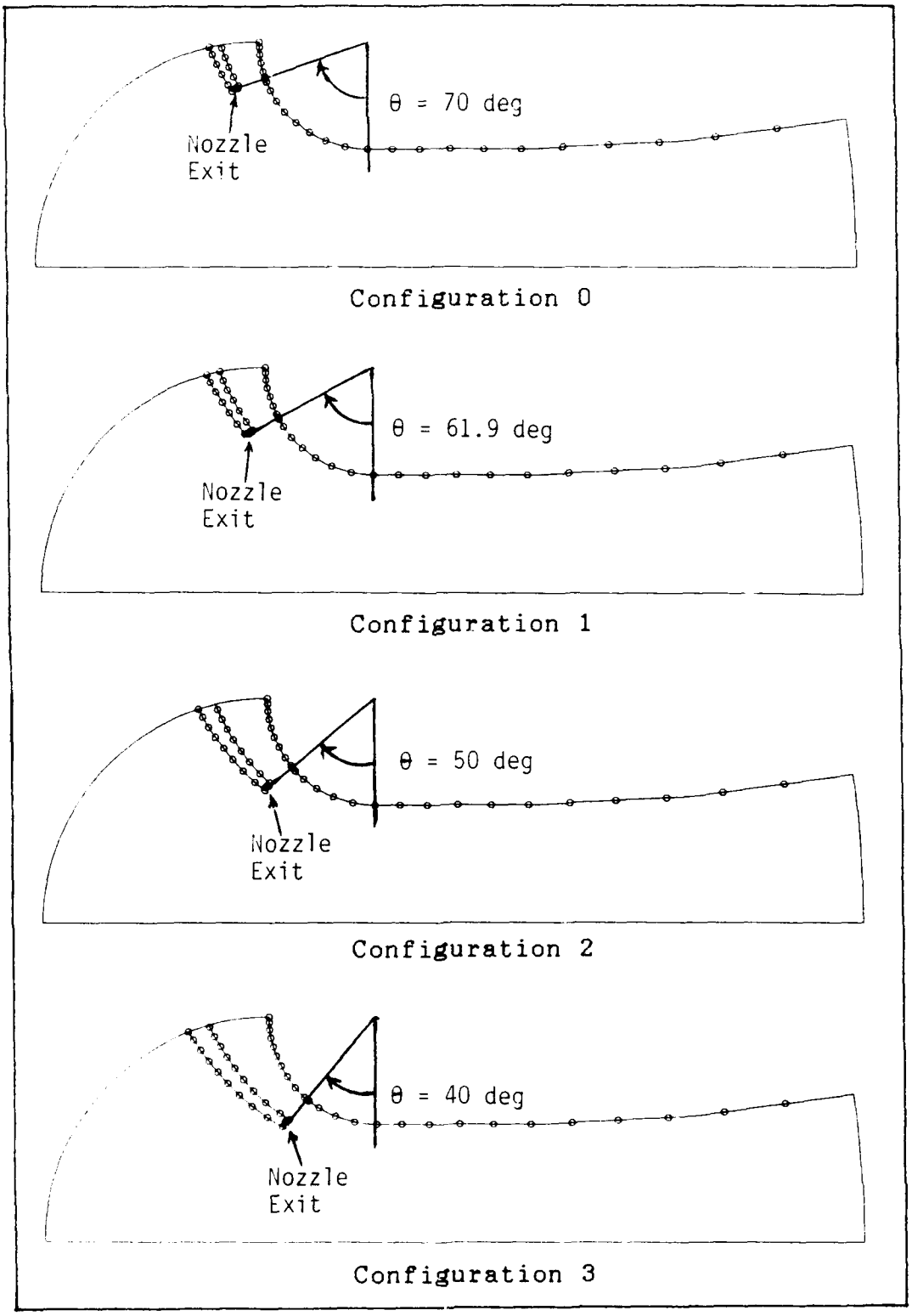

Fig. 13. Nozzle Locations

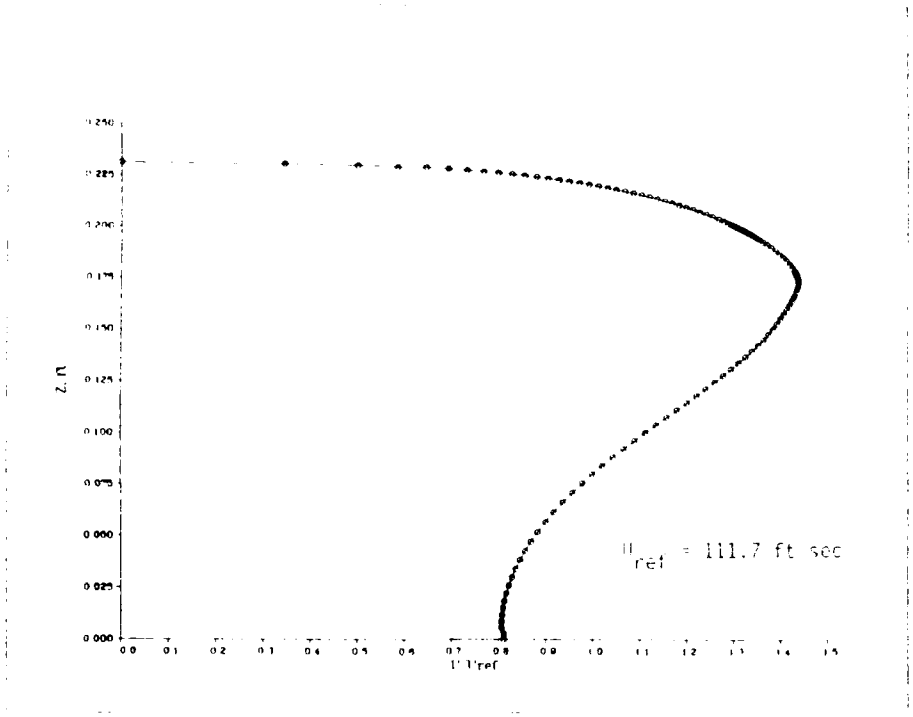

Fig. 14. Exit Velocity Profile - Numerical

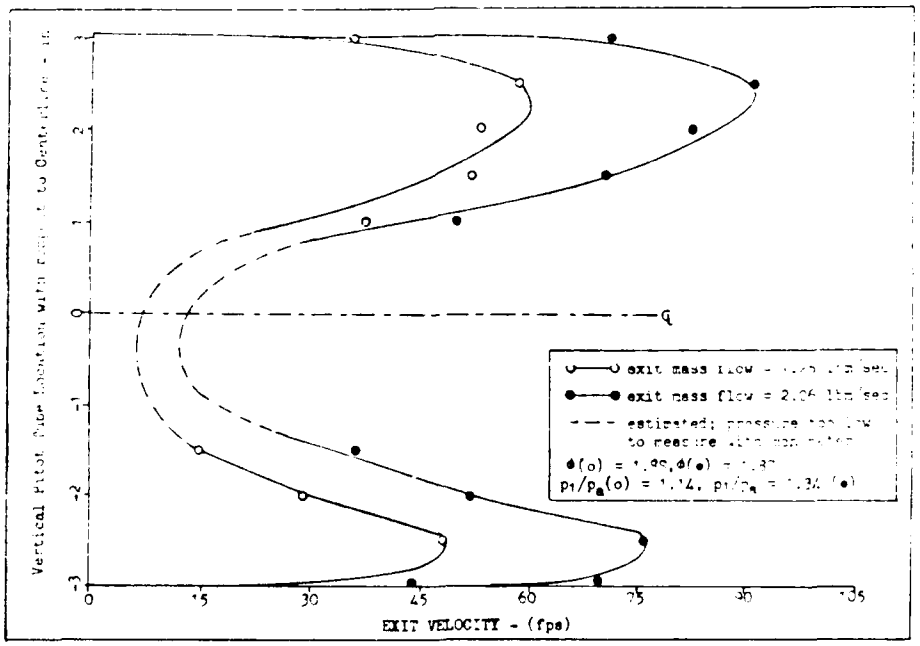

Fig. 15. Exit Velocity Profile - Experimental (11:48)

the shape of the profile is closely replicated by the model. This is an indication that the flow structure is indeed similar to that in the experimental ejector. However, since the experimental data was produced using the hook nozzles and is therefore only similar to the ejector modeled, the comparison cannot be exact.

Similarly, comparison of thrust augmentation vs. injection angle curves for the numerical model (Figure 16) with similar curves from Unnever (Figure 17) (17:46) show that the expected trends are indeed modeled. Again, since the experimental curves are from hook nozzle cases (by necessity, since the injection angle of the annular nozzle could not be varied) the comparison can only be qualitative, not quantitative.

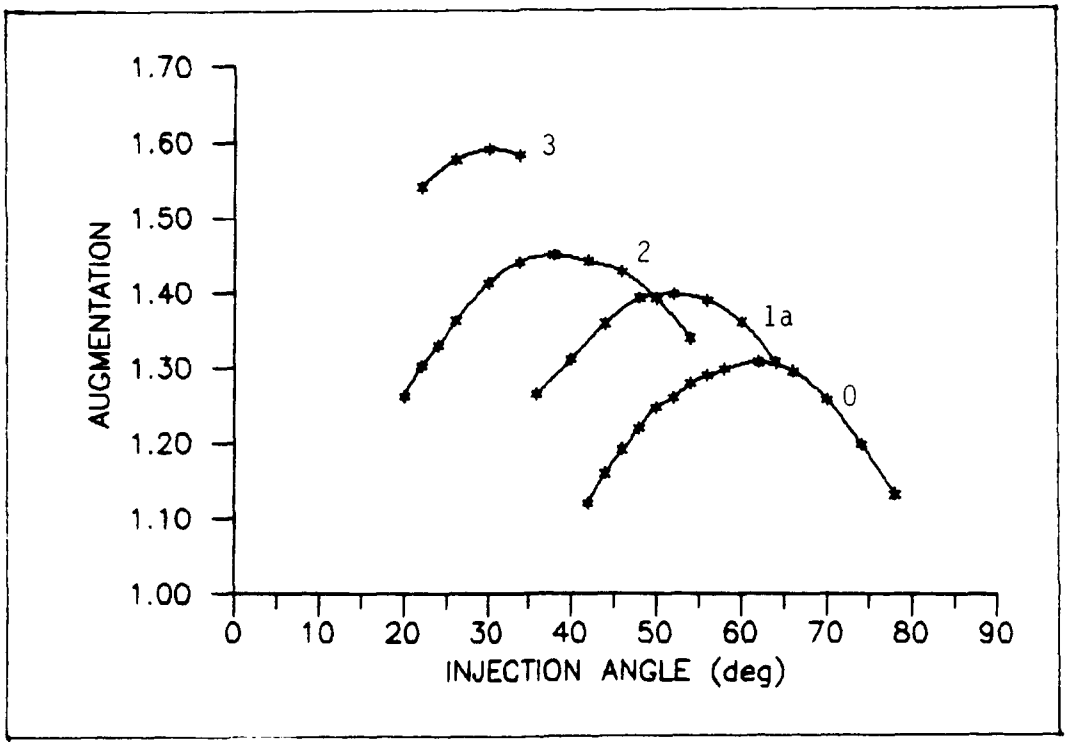

Fig. 16. Thrust Augmentation vs. Injection Angle Numerical

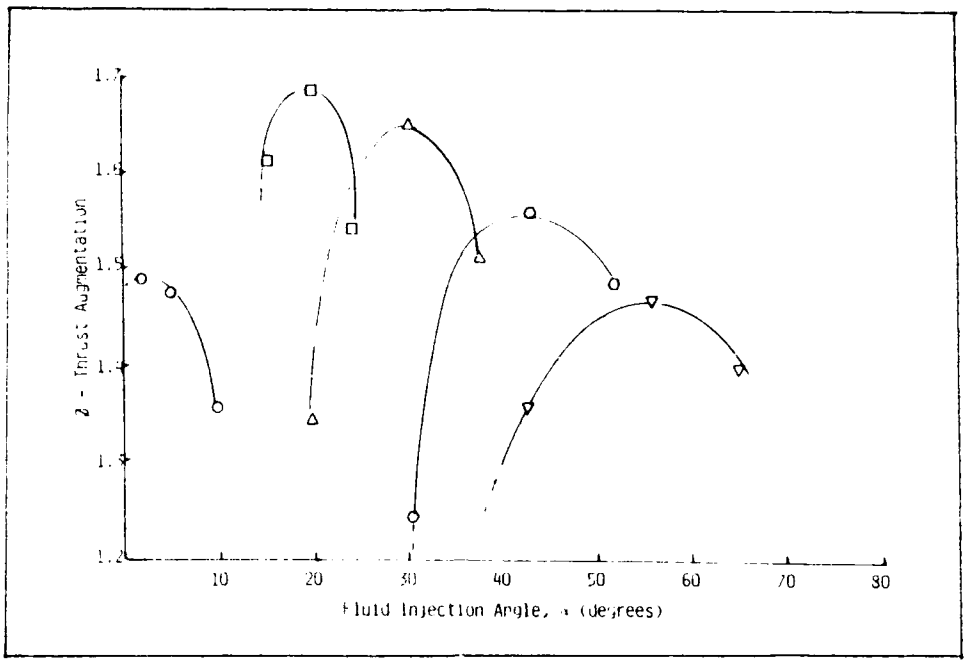

Fig. 17. Thrust Augmentation vs. Injection Angle Experimental

However, a quantitative comparison of the thrust augmentation from configuration 1a with Unnever's annular nozzle configuration (Table II) (17:60) shows that the numerical model did not achieve the thrust augmentation measured experimentally.

Table II
Thrust Augmentation Comparison

Ejector	Ø
Numerical Model	1.40
Experimental	1.48

Possible explanations for this difference include:

1. Turbulence modeling not optimized for a wall jet flow resulting in poorer mixing in the numerical model.

2. Three dimensional mixing or swirl in the experimental ejector which is not accounted for in the numerical model due to the assumption of axial symmetry.

3. Modeling the jet exiting the nozzle as perfectly parallel flow in the desired direction when there is likely some divergence in the actual jet experimentally.

Each of these lead to less complete mixing in the numerical model and would thus account for the lower thrust augmentation calculated. Additionally, other sources of error which would account for differences in experimental versus numerical results include:

4. Insufficient grid resolution.

5. Experimental error.

Using $\emptyset$ = 1.00, or no augmentation, as a baseline, this results in a 16.7 percent error. This error indicates that the model, as it is, would not be a good predictor of the actual thrust augmentation which could be expected from an experimental setup. However, since trends determined with this model appear to be consistent with experimental results, analysis of different parameters should allow identification of values which maximize thrust augmentation which could be used to guide hardware development.

Data Visualization. One of the benefits of a numerical simulation is that the solution results in complete knowledge of the computed flowfield. This knowledge can be

used to provide insight into the behavior of the flow within the ejector which would be nearly impossible to determine experimentally. Plotting velocity contours (Figure 18) clearly shows the primary characteristics of the flowfield.

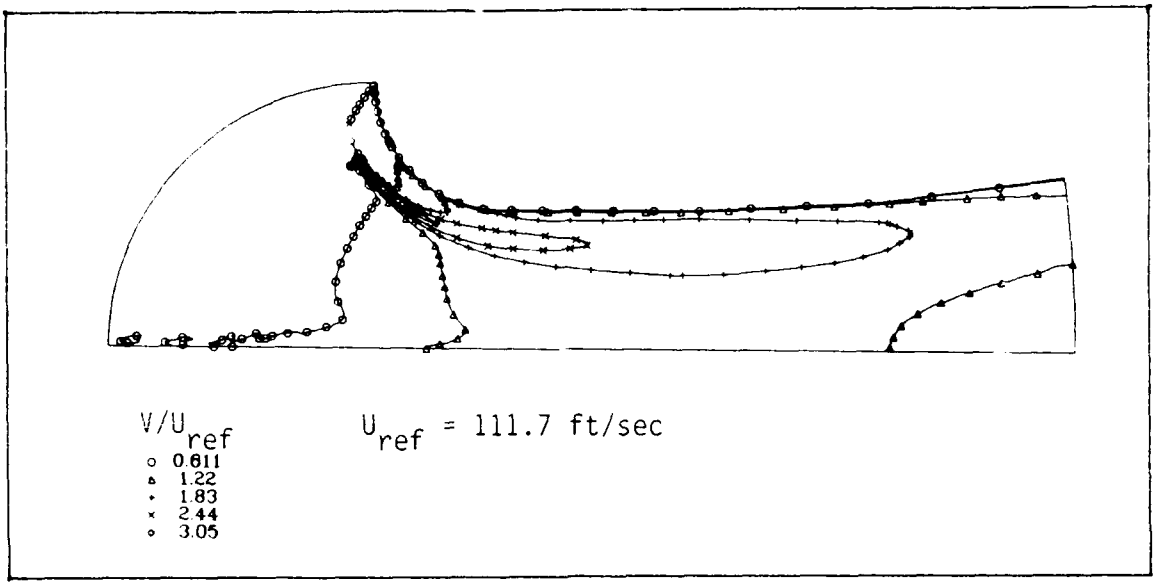

Fig. 18. Total Velocity Contours

First, (noting that the flow is from left to right) one can see how the jet hugs the wall and negotiates the curvature in the inlet due to the coanda effect. Second, one can see how the jet spreads out as it entrains more of the surrounding flow along with it. The increasing velocity of the entire flowfield as the cross-sectional area of the duct decreases can also be seen. Lastly, a relatively high velocity channel flow can be seen between the nozzle structure (and the jet) and the wall.

A streamline diagram of configuration 1a at its optimum injection angle (Figure 19) shows the smooth flow which characterizes the best performing injection angles.

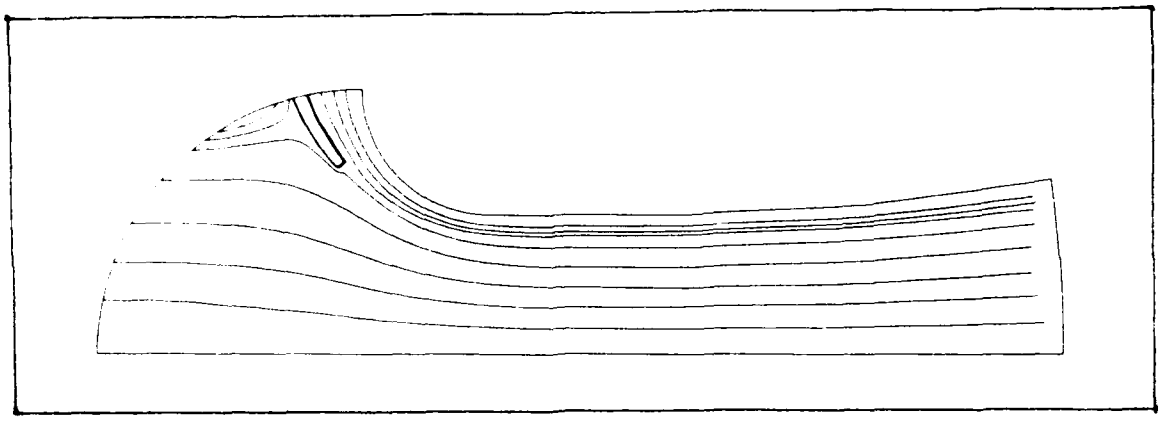

Fig. 19. Streamlines

One can also see, however, that a stagnation point exists on the nozzle structure upstream. Although the annular nozzle structure, as modeled, is thinner than that of the experimental apparatus, this diagram shows that significant flow blockage by the nozzle exists even in the numerical model just as was suspected in the experimental work by both Reznick and Unnever. A velocity vector plot of this region (Figure 20) clarifies the picture.

Finally, a sequence of plots of velocity profiles (Figures 21-23) shows how the jet mixes with the secondary flow as it progresses through the ejector. The first plot (Figure 21) shows the velocity profile at three grid stations ranging from the exit plane of the nozzle (K = 32) to a cross-section within the constant area mixing chamber, or throat, of the ejector (K = 75). These plots show clearly how the jet begins to mix with the secondary flow and how the entire flow accelerates as the inlet converges. It may further be determined from this plot is that the grid resolution at the upper surface appears to be barely sufficient to capture the boundary layer. However, it does not appear to be sufficient to capture boundary layer effects on the nozzle walls.

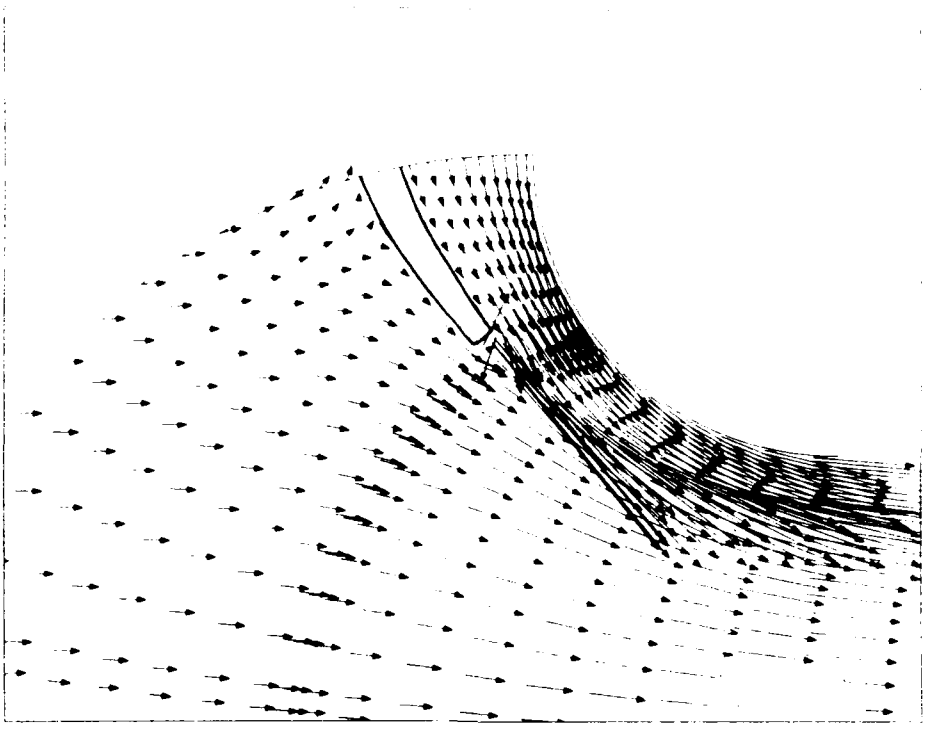

Fig. 20. Velocity Vectors - Nozzle Area

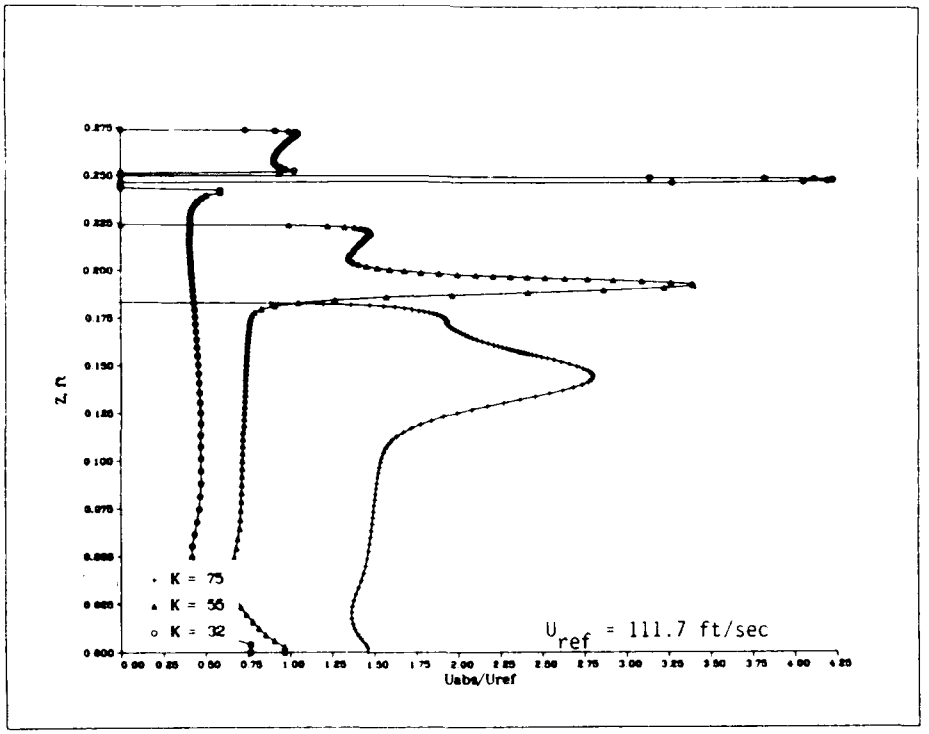

Fig. 21. Inlet Velocity Profiles

The second plot (Figure 22) shows how the jet continues to mix with the secondary flow as it progresses from the beginning (K = 75) to the end (K = 100) of the mixing chamber.

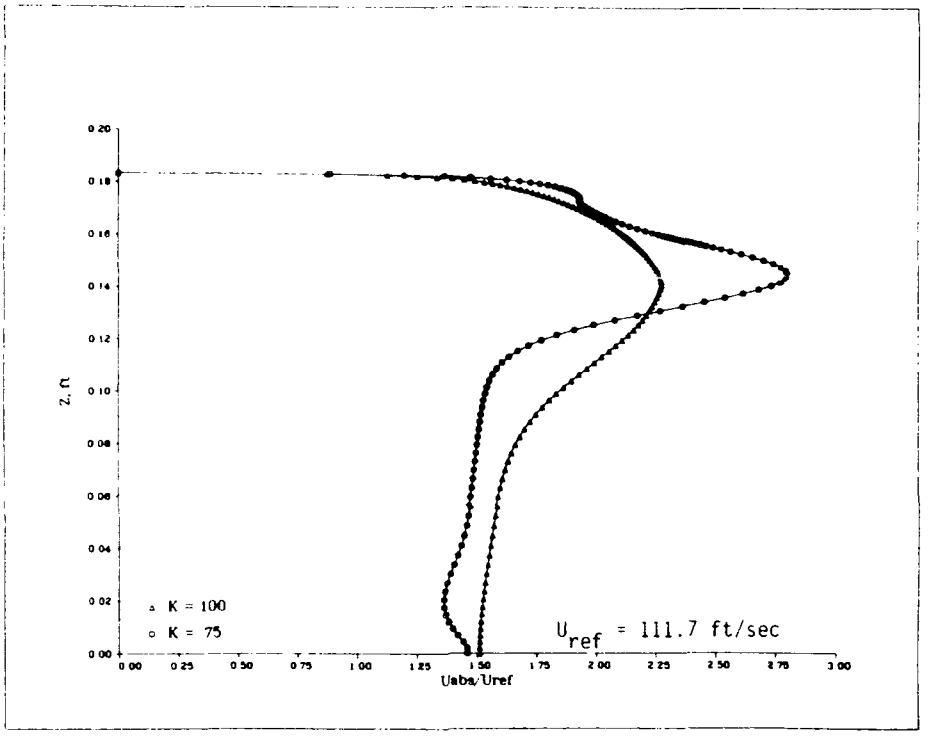

Fig. 22. Mixing Chamber Velocity Profiles

The last plot (Figure 23) portrays the changing profiles as the flow proceeds through the diffuser from the end of the mixing chamber (K = 100) to the exit (K = 130). The deceleration of the flow in the diffuser is evident. Also, the continued spreading of the jet as it mixes with the secondary flow is clearly visible. These basic flow features were consistent for all of the configurations modeled.

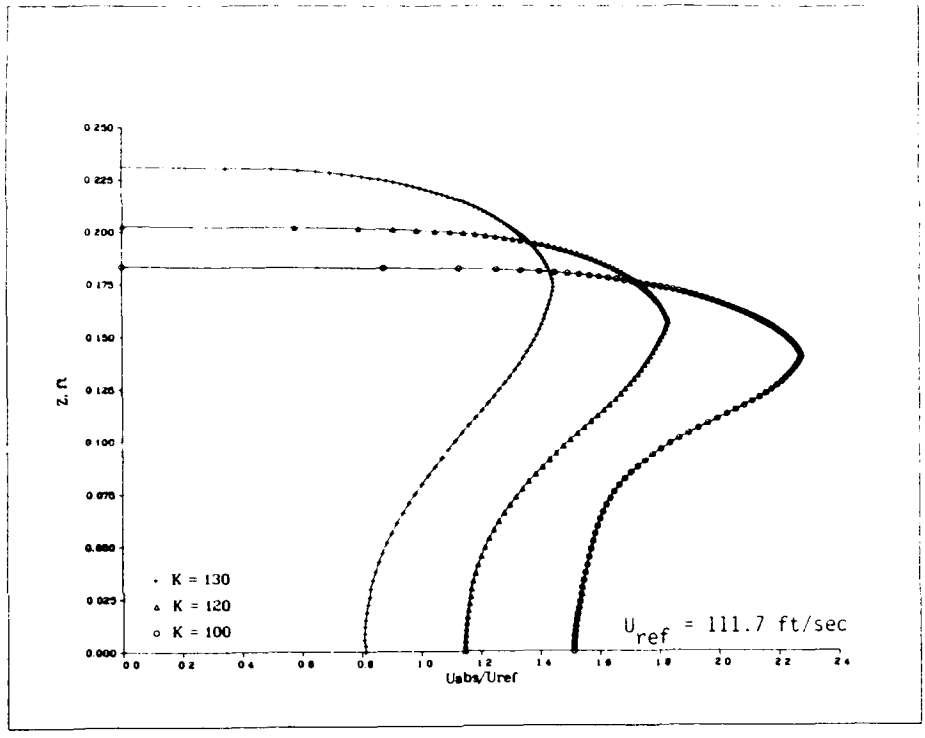

Fig. 23. Diffuser Velocity Profiles

Flexibility. This code, as modified for this investigation, is capable of handling variations of injection angle, nozzle position, nozzle gap (and therefore area) and pressure ratio. With some additional effort, diffuser and inlet geometries could be altered as well.

Secondary Objectives

Several of the secondary objectives were approached in the course of tnis investigation although all would require further work to provide the requisite parametrics to use for hardware design. The parameters studied include injector angle, nozzle position, and nozzle to throat area ratio.

Injection Angle. The effect of injection angle on the flowfield can be observed by comparing the optimum injection angle for configuration 1a to two off-nominal cases. For configuration 1a, the optimum injection angle was found to be α_1 = 52 degrees. This is compared to the off-nominal cases of α_1 = 64 degrees and α_1 = 36 degrees. The thrust augmentation achieved by each of these cases is tabulated in Table III.

Table III
Augmentation with Changing Injection Angle (α_1)

α_1	ϕ
36	1.264
52	1.398
64	1.308

First, looking at velocity contours for the α_1 = 64 degree case (Figure 24) one can see that the jet flow does not follow the wall closely but instead spreads out into the main mixing chamber. The improved mixing accomplished by the increased injection angle would seem to be desirable. But, as the exit velocity profiles indicate (Figure 25), the increased centerline flow is accomplished at the expense of a large drop in flow velocity near the wall. For a circular ejector, the region near the wall represents a much greater area in which momentum was sacrificed than the area near the centerline where momentum was gained. Also, it can be seen that the flow has begun to separate from the wall resulting in reversed flow.

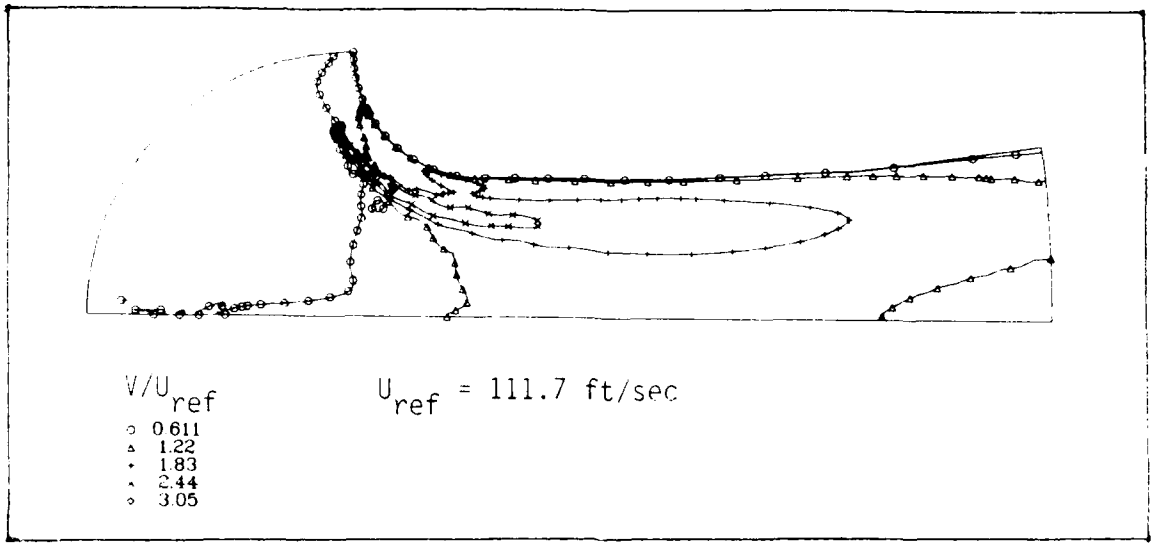

Fig. 24. Velocity Contours : α_1 = 64 degrees

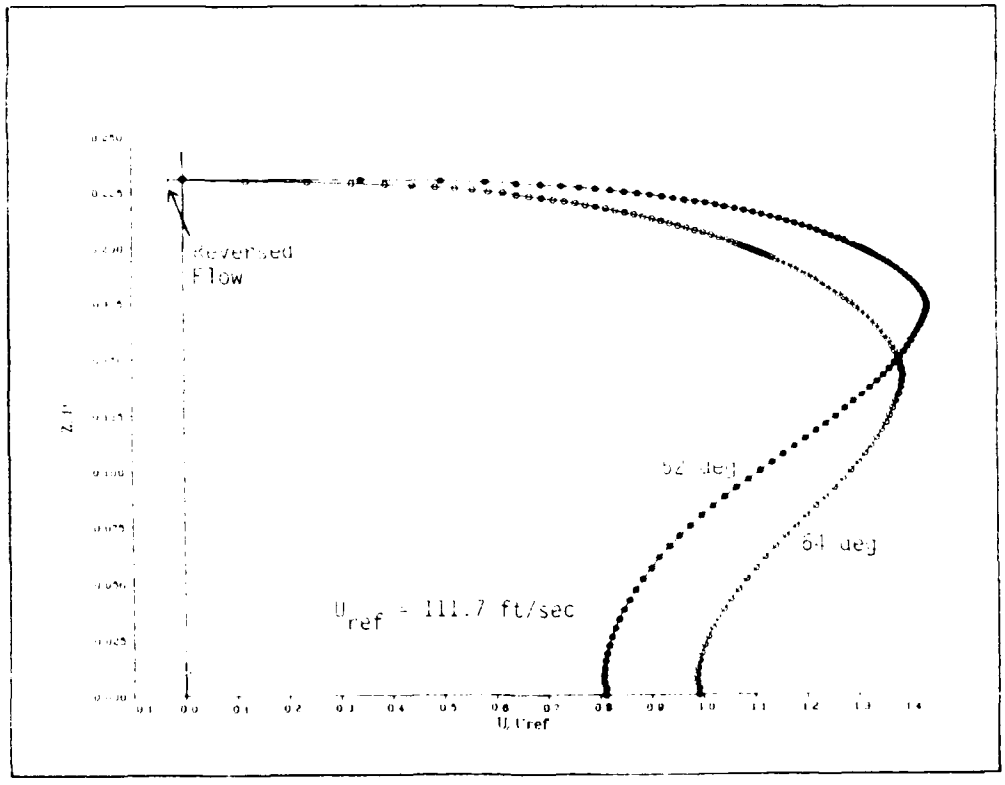

Fig. 25. Exit Velocity Profiles : α_1 = 52 and 64 degrees

The second off-nominal case (α_1 = 36 degrees) is one in which the jet is directed too near the wall. A first look at the velocity contours (Figure 26) shows the jet closely hugging the wall which by the above argument would seem to be desirable. However, a closer look at the area between the jet and the wall shows that, contrary to the optimum case (Figure 18), the flow through this channel is of relatively low velocity. Also, the high speed jet flow along the wall loses more momentum due to frictional effects.

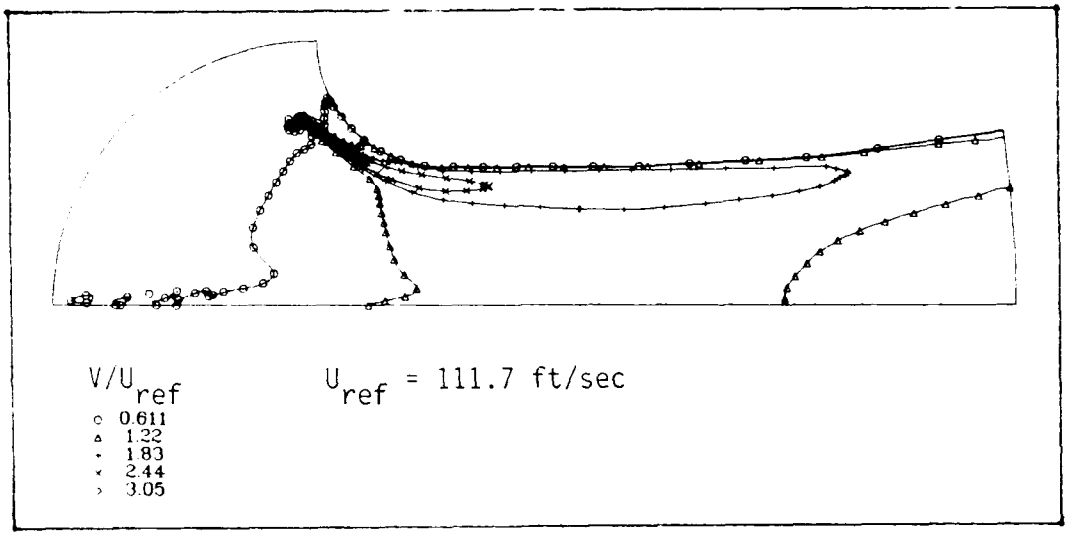

Fig. 26. Velocity Contours : α_1 = 36 degrees

The effect of the blockage of this high energy flow source and frictional losses are reflected in the exit profile (Figure 27). Although the jet is more concentrated for the α_1 = 36 degree case it is no more powerful than the optimal, α_1 = 52 degree, case. The reduced mixing also results in a decrease in momentum across the majority of the

center section of the ejector. Apparently, the optimum case is one which balances these considerations by achieving maximum mixing without allowing the jet to separate from the wall.

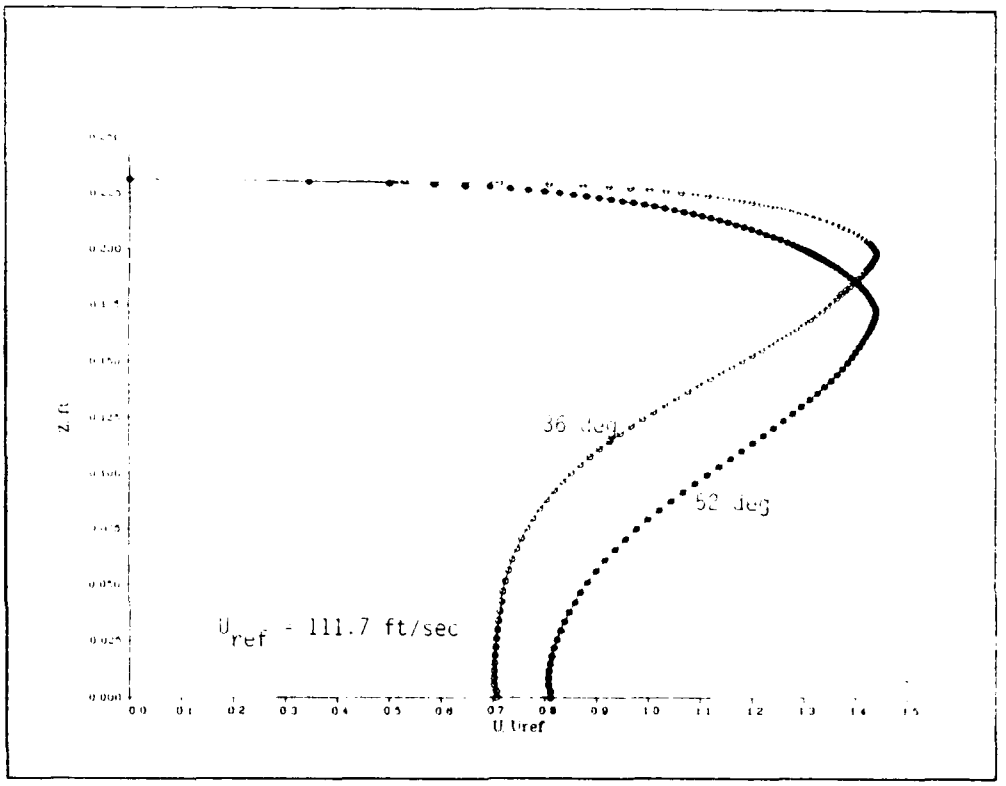

Fig. 27. Exit Velocity Profiles : i = 52 and 36 degrees

Figure 28 defines a pair of angles which the injection angle of the nozzle can be referenced to. These angles are those representing injection parallel and tangent to the wall. Figure 29 shows the relationship of the injection angle for maximum thrust augmentation to those representing injection parallel to the wall and tangent to the wall for each of the four configurations investigated. In all cases, the optimum injection angle is approximately a quarter of

the way between parallel and tangent to the wall. This conclusion must be caveated by noting that, due to the circular inlet lip and the fact that all configurations were limited to be the same distance from the wall, the relative geometry between the nozzle and the wall is the same for all configurations. To fully investigate this parameter, additional data would need to be collected at different distances from the wall and (if desired) for different inlet geometries.

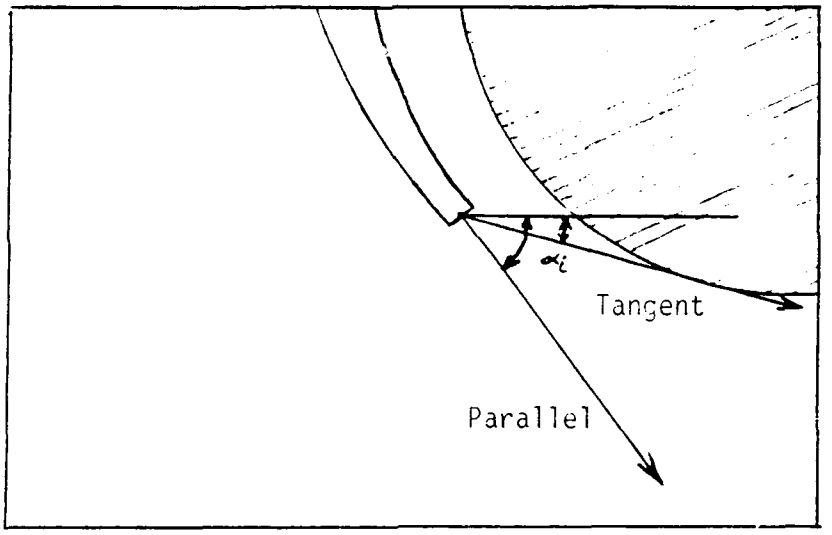

Fig. 28. Reference Injection Angles

Nozzle Position. In investigating the influence of nozzle position on thrust augmentation, it was necessary to isolate the effect of nozzle movement by constraining the configurations chosen. In order to eliminate changing wall effects, all configurations were set to have the nozzle the same distance from the wall as configuration 1a (0.55 in.).

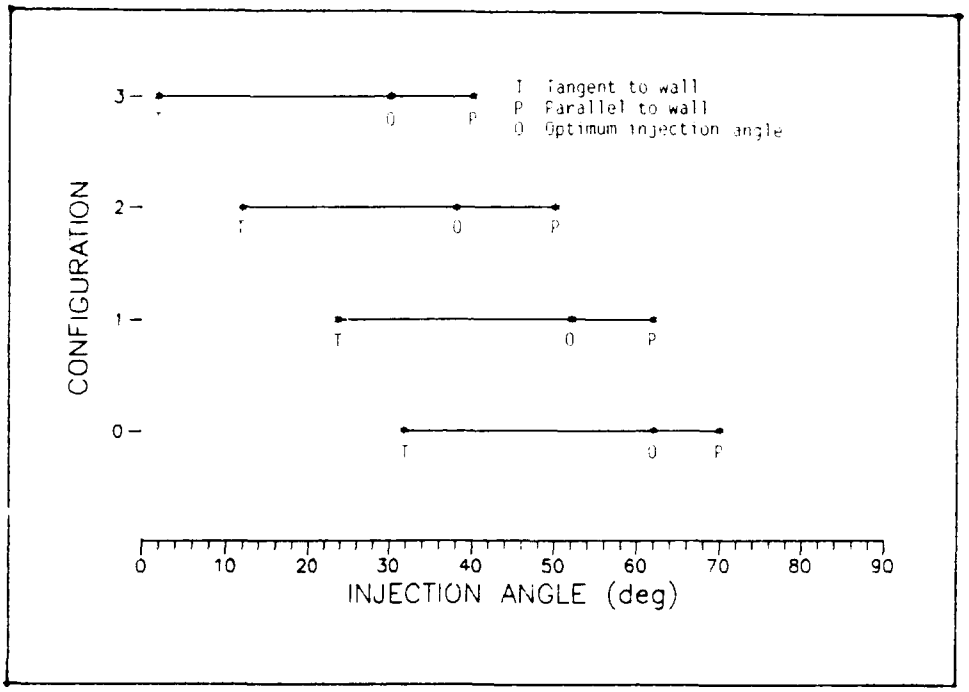

Fig. 29. Optimum Injection Angle with Relation to Reference Angles

The nozzle gap size was adjusted to achieve similar nozzle areas for all configurations to eliminate area ratio effects. This reduces the effect of location changes to only the effect of nozzle position angle, θ. Plotting the maximum augmentation achieved at each location vs. the position angle (Figure 30) a definite trend can be detected. Higher augmentation ratios are achieved with decreasing position angle, or in other words, placing the nozzle further into the inlet.

The effect of nozzle location can best be seen by comparing the two extremes studied, configuration 0 and configuration 3. By looking at velocity profiles of the jet itself (Figures 31 and 32) we can see that configuration 3 has a higher peak jet velocity than configuration 0.

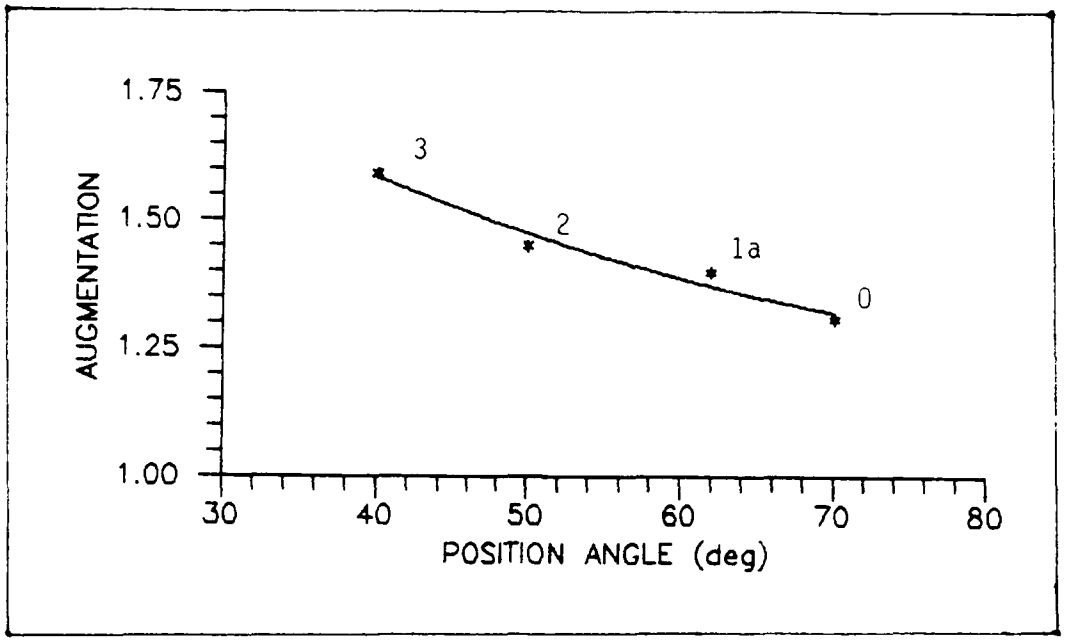

Fig. 30. Augmentation vs. Position Angle

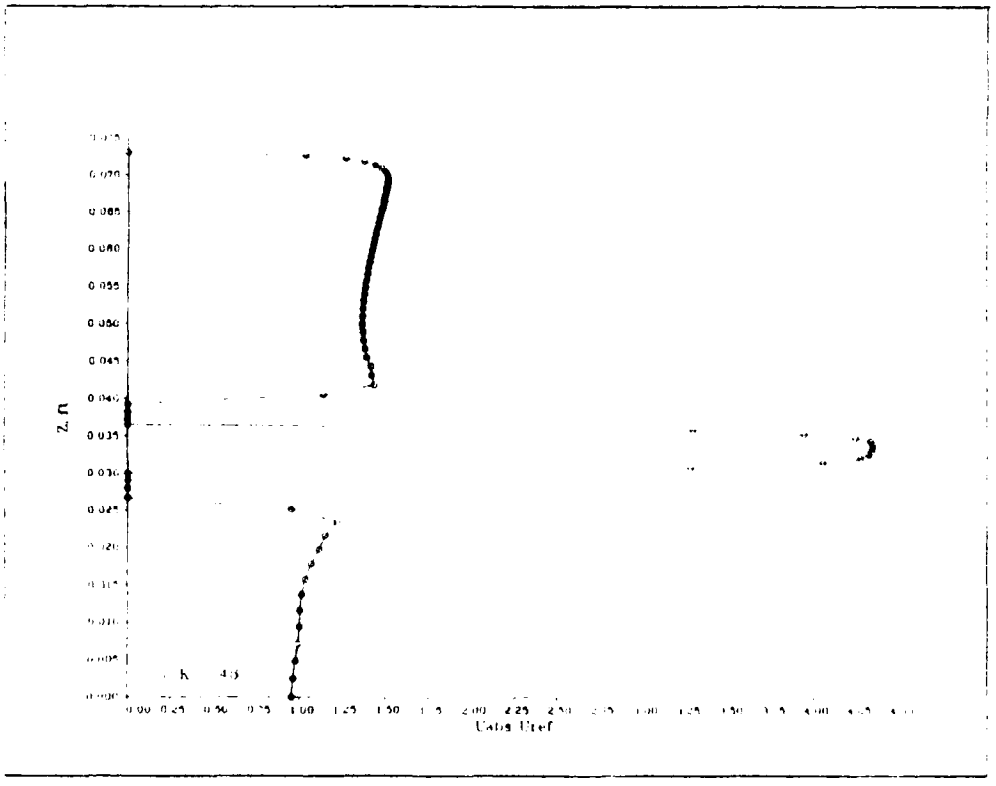

Fig. 31. Nozzle Velocity Profile : Configuration 3

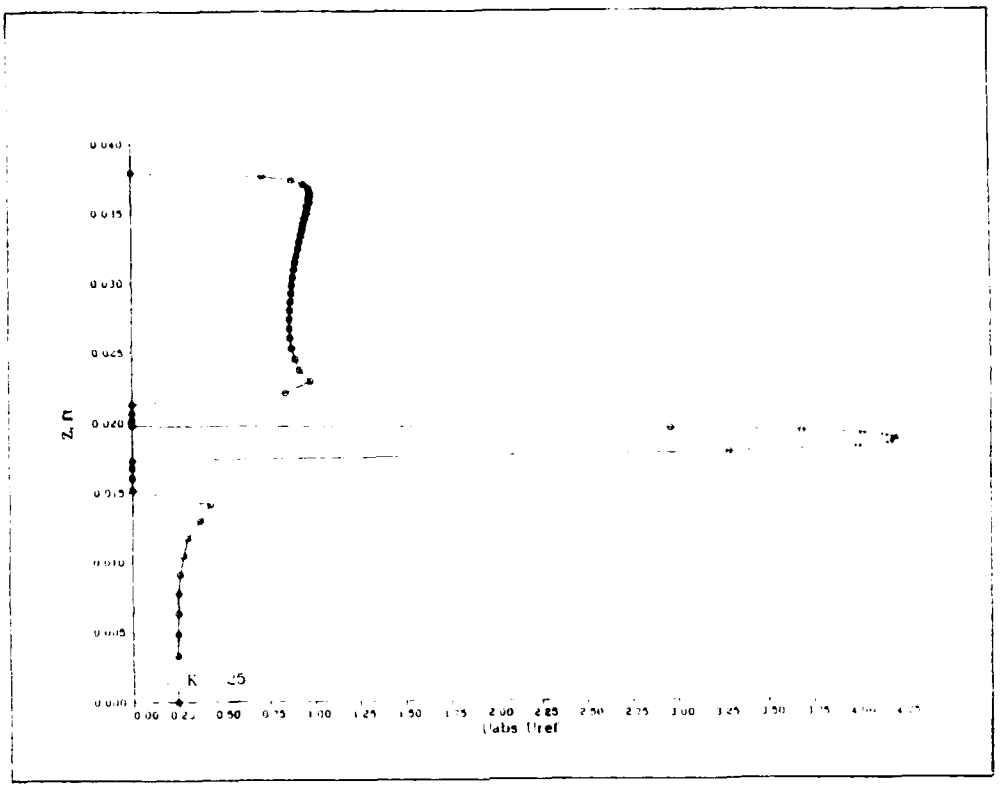

Fig. 32. Nozzle Velocity Profile : Configuration 0

This is due to the fact that the plenum pressure of both configurations is the same:

$$P_{o\ noz} = P_{rat}\ (P_a) \qquad (47)$$

but, as can be seen in Figures 33 and 34, the exit of configuration 3's nozzle is in the lower pressure region nearer the throat of the ejector. The pressure near the exit of configuration 3's nozzle (Figure 33) is about 0.990 P_a on the side nearer the wall and about 0.994 P_a on the opposite side. This is compared to configuration 0's nozzle (Figure 34) which is located where the pressures are approximately 0.996 P_a and P_a respectively. Since the exit

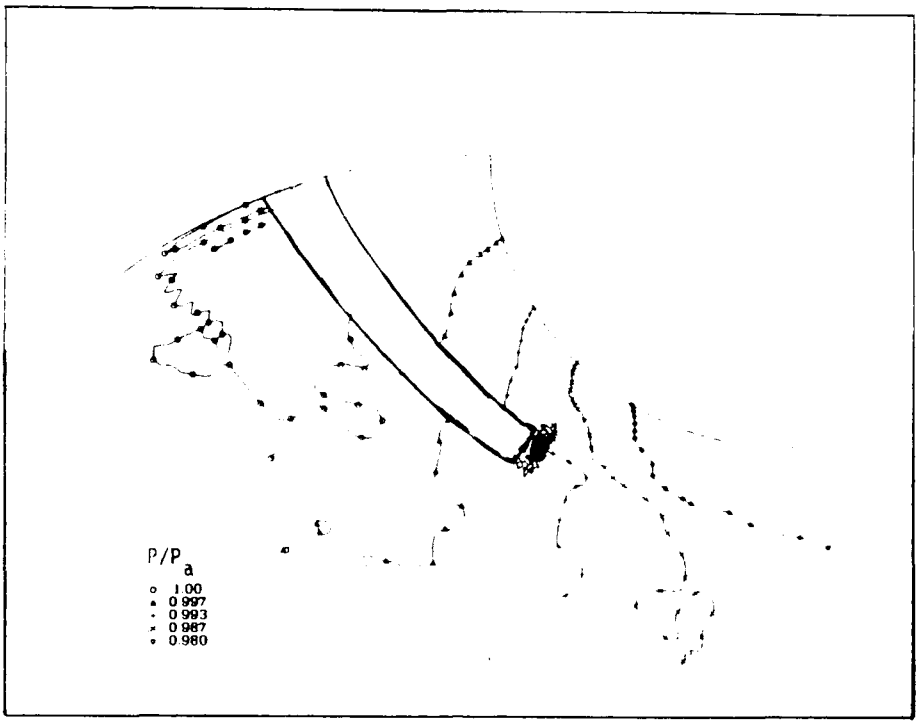

Fig. 33. Pressure Contours : Configuration 3

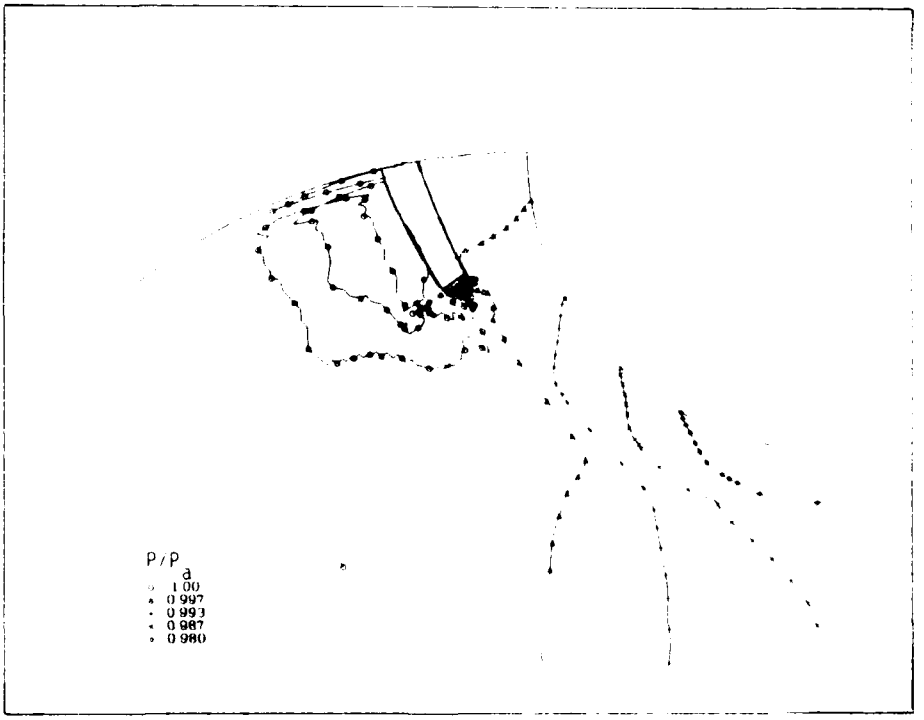

Fig. 34. Pressure Contours : Configuration 0

pressure is lower for configuration 3, the exit velocity is necessarily higher. Also, there is less momentum loss due to friction. This model does not, however, account for losses within the nozzle which would reduce the effectiveness of the longer nozzle structure of configuration 3. One would assume that there would be some point at which the advantages of moving the nozzle toward the throat would be outweighed by the reduced mixing length allowed. Further analysis would be necessary to define the optimum location. One can also see in these plots how a pressure difference is maintained across the jet it which forces the jet to turn. This is similar to the turning of the flow from a jet flap.

Nozzle Area Ratio. Configurations 1b and 1c were designed to keep all other factors the same as configuration 1a except the nozzle gap, and therefore the nozzle area, in order to investigate the effect of nozzle to throat area ratio changes. Configuration 1b had a reduced nozzle area and configuration 1c had an increased nozzle area relative to configuration 1a. The area ratios and achieved thrust augmentation were as in Table IV. The results are plotted in Figure 35.

Table IV

Augmentation with Changing Nozzle Area Ratio (α)

Config.	Nozzle Area Ratio	$\emptyset$
1b	0.0650	1.407
1a	0.0887	1.398
1c	0.1167	1.353

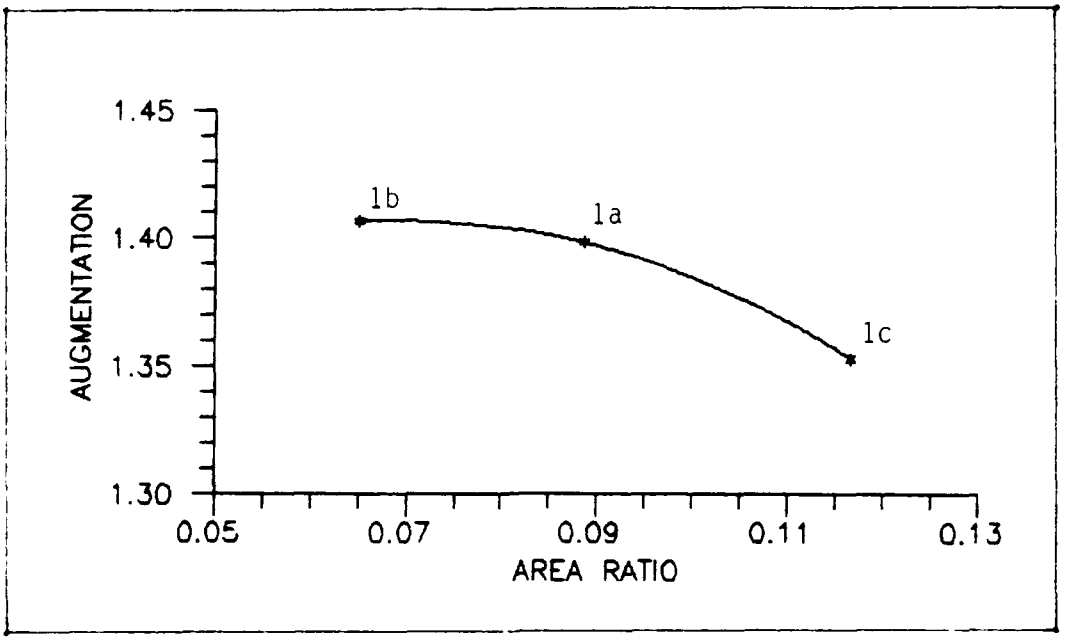

Fig. 35. Augmentation vs. Nozzle Area Ratio (α)

As can be seen from Figure 35, further reduction in area ratio would appear to be fruitless, contrary to the theoretical analysis presented earlier which predicted ever increasing augmentation with decreased area ratio. But, as Figure 36 shows, these results are in line with experimental results from Fought's MS Thesis as published in McCormick. These results suggest that the area ratio used by Reznick and Unnever (equivalent to configuration 1a) is near the optimum for this size ejector (Diameter = 112 mm).

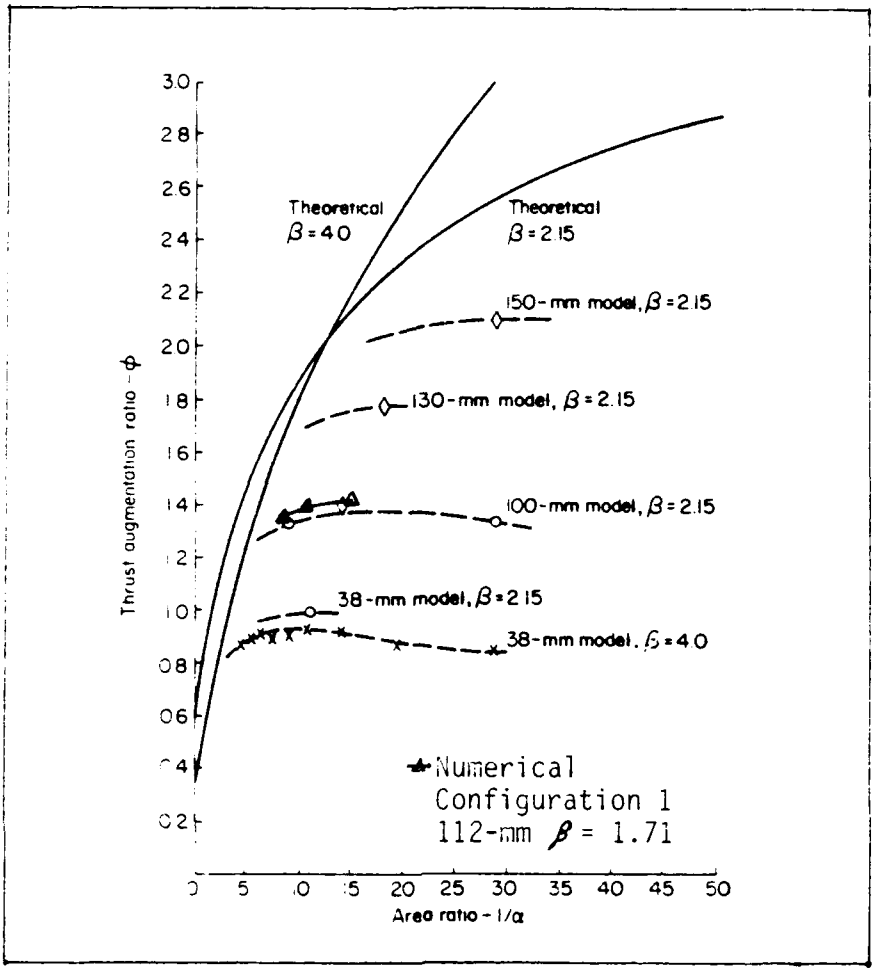

Fig. 36. Augmentation vs. Nozzle Area Ratio ($1/\alpha$) Numerical and Experimental (9:285)

IV. Conclusions

All primary objectives of this thesis have been been met. The numerical model has been shown to provide a reasonable simulation of the flow characteristics through the selected ejector configuration. Although the simulation is not sufficient to permit accurate prediction of actual thrust augmentation values for experimental hardware, all trends appear to be adequately modeled. Taking advantage of the complete knowledge of the flowfield provided by a numerical solution, the dynamics of the computed internal flow were able to be analyzed. Data visualization techniques were able to give us insight into the internal flow of the ejector which could not be measured experimentally. Sufficient flexibility has been incorporated in the code to handle variations of parameters such as injection angle, nozzle position, nozzle to throat area ratio, pressure ratio, as well as inlet and diffuser geometries.

In addition, the secondary objective of exploring the effect of these parameters on thrust augmentation was partially investigated allowing some initial conclusions to be drawn which could affect future experimental hardware design.

1) The area ratio used by Reznick and Unnever is near the optimum and should be sufficient for future experiments. However, if it is within manufacturing capability, some reduction of nozzle area would be beneficial.

2) Future experiments should concentrate on configurations which inject the jet at a location closer to the entrance of the constant area mixing chamber (smaller θ) if possible.

3) Initial indications are that an annular nozzle design should attempt to provide an injection angle approximately 1/4 of the way between parallel to the wall and tangent to the wall for the selected nozzle location.

V. Suggestions and Recommendations

Any continued research in this area should concentrate on expanding the parametric analysis in the areas of nozzle location as well as inlet and diffuser geometries. Nozzle locations further into the inlet and variations in distance from the wall should be investigated. Larger diffuser ratios should also provide increased augmentation capability and different inlet shapes could optimize the incoming flow. Further analysis of injection angle effects for the new geometries may also be necessary.

Another primary area of research which would be beneficial is application of improved turbulence modeling, better grid refinement, and second-order boundary conditions to improve the numerical model and try to bring the predicted thrust augmentation levels in line with experimental results. It may even be desirable to change to a different solution algorithm to reduce computation time.

Lastly, using the parametric results presented here to design new experimental hardware should result in improved performance and would serve to better validate this model.

Appendix A: Grid Generating Programs and Tables

Appendix A-1: Grid Generating Program *EJECORD*

```
                                                                      XKKS
      PROGRAM     EJECGRD
      PARAMETER   (JDIM = 130, LDIM = 115, JARC = 120)
      PARAMETER   (THRHT = 1.25, NAR = 101)
      COMMON  /ANFANG/  AA(JDIM, LDIM), AM(JDIM)
      COMMON  /AREAS/   AL, ANG(LDIM)
      COMMON  /ARATS/   AMR(NAR), ARAT(NAR)
      COMMON  /DAT/     L1, PSI1, L2, PSI2, XCYL, XCONE1
      COMMON  /FARCL/   PARC(LDIM)
      COMMON  /KSI/     AX(JDIM, LDIM), AY(JDIM, LDIM)
      COMMON  /NOZ/     JNOZIN, JNOZOUT
C
C        COMPUTE WALL ARC LENGTH VS X
C        J COUNTS HORIZONTALLY (AXIALLY)
C        L COUNTS VERTICALLY (RADIALLY)
C
      DIMENSION   ARCLEN(JARC), GS(JDIM), XARC(JARC), XP(LDIM),
     1  YP(LDIM)
      LOGICAL     SUBSON
      XCYL   = 3.0
      L1     = 3.5
      PSI1   = 0.05235987756
      L2     = 3.5
      PSI2   = 0.13962634
      XCONE1 = XCYL + L1 * COS(PSI1)
C
C        GENERATE TABLE OF ONE-DIMENSIONAL AREA RATIOS FOR
C        SUBSONIC MACH NUMBERS
C
      CALL ARF
C
C        GENERATE TABLE OF ARC LENGTH ALONG OUTER WALL
C
      CALL ARCDAT (ARCLEN, XARC, JARC)
C
C        READ VERTICAL DISTRIBUTION OF HORIZONTAL LINES
C
      CALL READ64
C
C        READ HORIZONTAL DISTRIBUTION OF VERTICAL LINES
C
      OPEN (7, FILE = 'EJECHRZ', STATUS = 'OLD')
      REWIND 7
      SUBSON    = .TRUE.
        DO 1 J    = 1, JDIM
        READ (7,  100) GS (J)
        S         = GS(J) * ARCLEN(JARC)
        XW        = THIXT(S, ARCLEN, XARC, JARC)
C
C         SUBROUTINES LEFT AND RIGHT DROP THE VERTICAL GRID LINES
C         FROM THE SPECIFIED WALL POSITION, XW.
C
        IF (XW .LE. 0.0) THEN
          CALL LEFT (XW, XP, YP)
        ELSE
          CALL RIGHT (XW, XP, YP)
        END IF
C
C         AL IS THE ARC LENGTH OF THE CURRENT VERTICAL GRID LINE
C
        AR        = (AL / THRHT) ** 2
        JNOZIN    = 57
        JNOZOUT   = 69
        CALL MARF (AR, SUBSON, AM (J))
          DO 1 L    = 1, LDIM
C
C         AX AND AY ARE THE GRID COORDINATES; SUBSCRIPTS INCREASE
C         DOWNWARD FROM THE WALL.
C         AA IS THE SLOPE OF THE HORIZONTAL GRID LINE.
C
          AX(J, L)  = XP(L)
          AY(J, L)  = YP(L)
          AA(J, L)  = ANG(L)
    1     CONTINUE
      CLOSE (7)
C
C         WRITE FILE FOR JSIAXK
C
      CALL FINAL
      STOP
C
  100 FORMAT  ( 2X, 4E14.7 )
C
      END
      SUBROUTINE  ARF
      PARAMETER   (GAM = 1.4, NAR = 101, DM = 0.01)
      COMMON  /ARATS/  AMR(NAR), ARAT(NAR)
C
      B         = 0.5 * (GAM + 1.0) / (GAM - 1.0)
      A         = (2.0 / (GAM + 1.0)) ** B
      C         = 0.5 * (GAM - 1.0)
      AMR(NAR)  = 0.00001
      ARAT(NAR) = (A / AMR(NAR)) * (1.0 + C * AMR(NAR) ** 2) ** B
        DO 1 I    = 2, NAR
        K         = NAR + 1 - I
        AMR(K)    = DM * FLOAT(I - 1)
        ARAT(K)   = (A / AMR(K)) * (1.0 + C * AMR(K) ** 2) ** B
    1   CONTINUE
      RETURN
      END
      SUBROUTINE  FINAL
      PARAMETER   (JDIM = 130, LDIM = 115, JARC = 120,
      COMMON  /ANFANG/  AA(JDIM, LDIM), AM(JDIM)
      COMMON  /KSI/     AX(JDIM, LDIM), AY(JDIM, LDIM)
      COMMON  /NOZ/     JNOZIN, JNOZOUT
C
      DIMENSION   RHO(JDIM, LDIM), RHOE(JDIM, LDIM),
     1  RHOU(JDIM, LDIM), RHOV(JDIM, LDIM), ZA(JDIM, LDIM), ZM(JDIM,
     2  ZX(JDIM, LDIM), ZZ(JDIM, LDIM)
      LOGICAL     LAMIN
C
      EQUIVALENCE (FSMACH, XM), (TW, TWALL)
C
      DATA    TO, PO / 518.7, 14.698 /
      DATA    TW / 518.7 /
      DATA    CV, RG / 4290.0, 1718 /
C
C     NAMELIST /JIX  ALPHA, BETA, CFL, CX, CY, ISMTHX, ISMTHY,
C    1  ISTART, JL, KL, KSTEP, LAMIN, NEND, PTIN, REY, RL, TINF,
C    2  TTIN, TW, XM
```

```
C
      ALPHA      = -1.0818
      BETA       = 0.0
      CFL        = 0.0050
      CX         = 3.0
      CY         = 3.0
      ISMTHX     = 1
      ISMTHY     = 1
      ISTART     = 0
      JL         = LDIM
      KL         = JDIM
      KSTEP      = 4
      LAMIN      = .FALSE.
      NEND       = 1
      PTIN       = 2116.5
      REY        = 341325
      RL         = 0.47838
      TINF       = 518.44
      TTIN       = 518.7
      TW         = 518.7
      XM         = 0.100
C
C     CONVERT TO FEET, PSF, ETC.
C
      PT         = 144.0 * PO
C
C     FLIP SUBSCRIPTS OF VERTICAL LINES END-FOR-END
C
      DO 1 L     = LDIM, 1, -1
      N          = LDIM + 1 - L
        DO 1 J   = 1, JDIM
        ZX(J, L) = AX(J, N)
        ZZ(J, L) = AY(J, N)
        ZA(J, L) = AA(J, N)
C       ZM(J)    = AM(J)
1     CONTINUE
C
C     EXIT CONDITIONS
C
      XM         = AM(JDIM)
      TR         = 1.0 + 0.2 * XM ** 2
      TINF       = TO / TR
      VISC       = 2.27E - 08 * SQRT(TINF ** 3) / (TINF + 198.6)
      PINF       = PT / TR ** 3.5
      RHOINF     = PINF / (RG * TINF)
      RL         = ZZ(JDIM, LDIM) / 12.0
      CINF       = SQRT(1.4 * RG * TINF)
      UINF       = XM * CINF
      REY        = UINF * RL * RHOINF / VISC
C
      DO 2 J     = 1, JDIM
        DO 2 L   = 1, LDIM
        IF ((L .GE. JNOZIN) .AND. (L .LE. JNOZOUT)) THEN
          ZM(J) = 0.367
        ELSE
          ZM(J) = AM(J)
        ENDIF
          TR     = 1.0 + 0.2 * ZM(J) ** 2
          T      = TO / TR
          CS     = 49.0428 * SQRT(T)
          RHO(J, L) = P / (RG * T)
          RHW    = P / (RG * TW)
          VEL    = CS * ZM(J)
          U      = VEL * COS(ZA(J, L))
          V      = VEL * SIN(ZA(J, L))
          IF (L .EQ. 1) THEN
            RHOU(J, L) = RHO(J, L) * U
            RHOV(J, L) = 0.0
            RHOE(J, L) = RHO(J, L) * (CV * T + 0.5 * VEL ** 2)
          ELSE IF (L .EQ. LDIM) THEN
            RHOU(J, L) = 0.0
            RHOV(J, L) = 0.0
            RHOE(J, L) = RHW * CV * TW
          ELSE
            RHOU(J, L) = RHO(J, L) * U
            RHOV(J, L) = RHO(J, L) * V
            RHOE(J, L) = RHO(J, L) * (CV * T + 0.5 * VEL ** 2)
          END IF
          ZX(J, L)   = ZX(J, L) / 12.0
          ZZ(J, L)   = ZZ(J, L) / 12.0
2     CONTINUE
C
      CALL X2DI (BETA, CFL, CX, CY, ISMTHX, ISMTHY, ISTART,
     1  JL, KL, KSTEP, LAMIN, NEND, REY, RL, TINF, TW, XM)
C
      OPEN (7, FILE = 'EJECO', FORM = 'UNFORMATTED', STATUS = 'NEW')
      WRITE (7)  ((ZX(J, L), J = 1, JDIM), L = 1, LDIM),
     1           ((ZZ(J, L), J = 1, JDIM), L = 1, LDIM)
      WRITE (7)  ((RHO(J, L), J = 1, JDIM), L = 1, LDIM),
     1           ((RHOU(J, L), J = 1, JDIM), L = 1, LDIM),
     2           ((RHOV(J, L), J = 1, JDIM), L = 1, LDIM),
     3           ((RHOE(J, L), J = 1, JDIM), L = 1, LDIM)
      CLOSE (7)
C
      RETURN
      END
      SUBROUTINE  LEFT (XW, XP, YP)
      PARAMETER   (LDIM = 115, LM = LDIM-1)
      COMMON  /CIRARC/  ALFA, ARCTOT, R, XC
      COMMON  /FARCL/  PARC(LDIM)
      COMMON  /AREAS/  AL, ANG(LDIM)
C
C     WORKING WITH CIRCLES CENTERED AT (XC, 0), NORMAL TO WALL
C     AT AT (XW, YW), AND TO CENTERLINE AT (XL, 0)
C     SUBSCRIPTS HERE RUN FROM TOP DOWN, ARE INVERTED LATER IN
C     SUBROUTINE  FINAL.
C
      DIMENSION   XP(LDIM), YP(LDIM)
C
      XWU        = XW
      CALL UPRLFT (XWU, XL, YWU, DYDX)
      XP(1)      = XWU
      YP(1)      = YWU
      ANG(1)     = ATAN(DYDX)
      XP(LDIM)   = XL
      YP(LDIM)   = 0.0
      ANG(LDIM)  = 0.0
C
        DO 1 L   = 2, LM
        ALF      = ALFA * (1.0 - PARC(L))
        YP(L)    = R * SIN(ALF)
```

```
      XP(L)       = XC - R * COS(ALF)
      SLNORM      = YP(L) / (XP(L) - XC)
      ANG(L)      = ATAN(SLNORM)
    1 CONTINUE
      AL          = ARCTOT
      RETURN
      END
      SUBROUTINE  MARF (AR, SUBSON, M)
      PARAMETER   (GAM = 1.4, TOL = 1.0E-6, NAR = 101)
      COMMON  /ARATS/  AMR(NAR), ARAT(NAR)
C
      LOGICAL     SUBSON
C
      B           = 0.5 * (GAM + 1.0) / (GAM - 1.0)
      E           = B - 1.0
      A           = (2.0 / (GAM + 1.0)) ** B
      C           = 0.5 * (GAM - 1.0)
      D           = (GAM - 1.0) * A * B
      IF (SUBSON) THEN
        M         = TWEEN(AR, ARAT, AMR, NAR)
      ELSE
        M         = 5.5
      END IF
      M0          = M
C
      L           = 0
    1 L           = L + 1
      T           = 1.0 + C * M * M
      AA          = (A / M) * T ** B
      F           = AA - AR
      FP          = D * T ** E - AA / M
      Y           = M - F / FP
      IF (ABS(Y-M) .GT. TOL) THEN
        M         = Y
        IF (L .LT. 20) GO TO 1
      END IF
      M           = Y
      IF (SUBSON .AND. M .GE. 1.0) M = M0
      IF (M .LE. 0.0) M = M0
      RETURN
      END
      SUBROUTINE  READ64
      PARAMETER   (LDIM = 115)
      COMMON  /FARCL/  PARC(LDIM)
C
      OPEN (64, FILE = 'EJECVRT', FORM = 'FORMATTED', STATUS = 'OLD')
        DO 1 L    = 1, LDIM
        READ (64,  100) PARC(L)
    1   CONTINUE
      RETURN
C
  100 FORMAT  ( 2X, E14.7 )
C
      END
      SUBROUTINE  RIGHT (XW, XP, YP)
      PARAMETER   (LDIM = 115, LM = LDIM-1)
      COMMON  /PBLARC/  SL, ARCTOT, S2L, S4L, Q4SL
      COMMON  /FARCL/   PARC(LDIM)
      COMMON  /AREAS/   TL, ANG(LDIM)
C
C         WORKING WITH PARABOLAS OF FORM X = A * Y ** 2 + C
C         THE SL BELOW IS A, XL IS C.
C
      XP(1)       = XW
      CALL UPPER (XP(1), YP(1), DYDX)
      ANG(1)      = ATAN(DYDX)
      SL          =  - 0.5 * DYDX / YP(1)
      XL          = XW - SL * YP(1) ** 2
      CALL TARCL (YP(1))
      TL          = ARCTOT
      XP(LDIM)    = XL
      YP(LDIM)    = 0.0
      ANG(LDIM)   = 0.0
        DO 1 J    = 2, LM
        YP(J)     = YP(1) * (1.0 - PARC(J))
        ARC       = ARCTOT * (1.0 - PARC(J))
        CALL PARCL (YP(J), ARC, SLNORM)
        ANG(J)    = ATAN(SLNORM)
        XP(J)     = XL + SL * YP(J) ** 2
    1   CONTINUE
      RETURN
      END
      SUBROUTINE  TARCL (Y11)
      COMMON  /PBLARC/  SL, ARCTOT, S2L, S4L, Q4SL
C
      DATA    EPS / 1.0E - 6 /
C
      IF (ABS(SL) .LE. EPS) THEN
        ARCTOT    = Y11
        RETURN
      END IF
      S2L         = 2.0 * SL
      S4L         = 4.0 * SL * SL
      Q4SL        = 0.25 / SL
      GP          = SQRT(S4L * Y11 * Y11 + 1.0)
      ARCTOT      = Q4SL * (S2L * Y11 * GP + LOG(S2L * Y11 + GP))
      RETURN
C
      ENTRY  PARCL (YG, ARC, SLNORM)
C
      Y1          = YG
      Y2          = YG
      IF (ABS(SL) .LE. EPS) GO TO 2
      J           = 0
    1 GP          = SQRT(S4L * Y1 * Y1 + 1.0)
      GY          = Q4SL * (S2L * Y1 * GP + LOG(S2L * Y1 + GP)) - ARC
      Y2          = Y1 - GY / GP
      J           = J + 1
      IF (ABS(Y2-Y1) .LT. EPS) GO TO 2
      Y1          = Y2
      IF (J .LE. 50) GO TO 1
      PRINT *, 'PARCL FAILED.'
    2 YG          = Y2
C
C         SLOPE OF NORMAL TO PARABOLA, IE, QUASI-STREAMLINE
C
      SLNORM      =  - S2L * YG
      RETURN
      END
      FUNCTION    TWIXT (X, XV, YV, N
```

```
C
C        SECOND - ORDER INTERPOLATION OF Y FROM TABLES OF
C        XV, YV WITH X AS ARGUMENT.
C
      DIMENSION   XV(N), YV(N)
C
      MID       = N / 2
      IF (X .GT. XV(MID)) GO TO 1
      KB        = 1
      KE        = MID
      GO TO 2
    1 KB        = MID
      KE        = N - 1
    2   DO 3 I    = KB, KE
        IF (X .GE. XV(I) .AND. X .LE. XV(I+1)) GO TO 4
    3   CONTINUE
C
C        ERROR... X NOT IN TABLE.
C
      THIXT     =  - 1.0
      RETURN
C
    4 KB        = I - 1
      IF (KB .GE. 1) GO TO 5
      KB        = 1
      KM        = 2
      KE        = 3
      GO TO 6
    5 KM        = I
      KE        = I + 1
    6 THIXT     = YV(KB) * (X - XV(KM)) * (X - XV(KE)) / ((XV(KB) -
     1  XV(KM)) * (XV(KB) - XV(KE))) + YV(KM) * (X - XV(KB)) * (X -
     2  XV(KE)) / ((XV(KM) - XV(KB)) * (XV(KM) - XV(KE))) + YV(KE) * (X
     3  - XV(KB)) * (X - XV(KM)) / ((XV(KE) - XV(KB)) * (XV(KE) -
     4  XV(KM)))
      RETURN
      END
      FUNCTION    TWEEN (X, XV, YV, N)
C
C        FIRST - ORDER INTERPOLATION OF Y FROM TABLES OF
C        XV, YV WITH X AS ARGUMENT.
C
      DIMENSION   XV(N), YV(N)
C
      MID       = N / 2
      IF (X .GT. XV(MID)) GO TO 1
      KB        = 1
      KE        = MID
      GO TO 2
    1 KB        = MID
      KE        = N - 1
    2   DO 3 I    = KB, KE
        IF (X .GE. XV(I) .AND. X .LE. XV(I+1)) GO TO 4
    3   CONTINUE
C
C        ERROR... X NOT IN TABLE.
C
      TWEEN     =  - 1.0
      RETURN
C
      IF (KB .GE. 1) GO TO 5
      KB        = 1
      KM        = 2
      KE        = 3
      GO TO 6
    5 KM        = I
      KE        = I + 1
    6 TWEEN     = YV(KM) + (X - XV(KM)) / (XV(KE) - XV(KM)) *
     1  (YV(KE) - YV(KM))
      RETURN
      END
      SUBROUTINE  UPPER (X, Y, DYDX)
      COMMON  /DAT/  L1, PSI1, L2, PSI2, XCYL, XCONE1
C
      YINT1  = 2.2 - XCYL * TAN(PSI1)
      YCONE1 = XCONE1 * TAN(PSI1) + YINT1
      YINT2  = YCONE1 - XCONE1 * TAN(PSI2)
      IF (X .LE. 0.0) THEN
        ARG       = SQRT(4.0 - X * X)
        Y         = 4.2 - ARG
        IF (ABS(ARG) .LE. 1.0E-6) THEN
          DYDX      =  - 1.0E4
        ELSE
          DYDX      = X / ARG
        END IF
      ELSE IF (X .LE. XCYL) THEN
        Y         = 2.2
        DYDX      = 0.0
      ELSE IF (X .LT. XCONE1) THEN
        DYDX      = TAN(PSI1)
        Y         = X * DYDX + YINT1
      ELSE
        DYDX      = TAN(PSI2)
        Y         = X * DYDX + YINT2
      END IF
      RETURN
      END
      SUBROUTINE  UPRLFT (XH, XL, YH, DYDX)
      COMMON  /CIRARC/  ALFA, ARCTOT, R, XC
C
      CALL UPPER (XH, YH, DYDX)
      B         = YH - DYDX * XH
      XC        =  - B / DYDX
      DELX      = XC - XH
      RSQ       = DELX ** 2 + YH ** 2
      R         = SQRT(RSQ)
      IF (ABS(DELX) .GT. 1.0E-6) THEN
        ALFA      = ATAN(YH / DELX)
      ELSE
        ALFA      = 1.570796
      END IF
      ARCTOT    = R * ALFA
      XL        = XC - R
      RETURN
      END
      SUBROUTINE  X2DI (BETA, CFL, CX, CY, ISMTHX, ISMTHY, ISTART,
     1  JL, KL, KSTEP, LAMIN, NEND, REY, RL, TINF, TW, XM)
C
      LOGICAL     LAMIN
C
      OPEN (7, FILE = 'JIX', STATUS = 'NEW')
```

```
      WRITE (7, 100) BETA, CFL, CX, CY, ISMTHX, ISMTHY, ISTART,
     1   JL, KL, KSTEP, LAMIN, NEND, REY, RL, TINF, TH, XM
      CLOSE (7)
      RETURN
C
  100 FORMAT ( ' $JIX' / ' BETA   =', F4.1, ',' /
     1   ' CFL    =', F7.4, ',' /
     2   ' CX     =', F4.1, ',' / ' CY     =', F4.1, ',' /
     3   ' ISMTHX =', I2, ',' / ' ISMTHY =', I2, ',' /
     4   ' ISTART =', I2, ',' /
     5   ' JL     =', I3, ',' / ' KL     =', I3, ',' /
     6   ' KSTEP  =', I2, ',' / ' LAMIN  =', L1, ',' /
     7   ' NEND   =', I6, ',' / ' REY    =', G12.5, ',' /
     8   ' RL     =', F7.4, ',' / ' TINF   =', F6.1, ',' /
     9   ' TH     =', F6.1, ',' / ' XM     =', F5.2, ',' /
     $   ' $END' )
C
      END
      SUBROUTINE  ARCDAT (ARCLEN, XARC, JARC)
      COMMON  /DAT/  L1, PSI1, L2, PSI2, XCYL, XCONE1
C
C       COMPUTE TABLE OF ARCLEN VS XARC
C
      DIMENSION   ARCLEN(JARC), XARC(JARC)
C
      ARCLEN(1) = 0.0
      XARC(1)   =  - 2.0
      ARCLEN(JARC) = 6.14159265 + L1 + L2
      XARC(JARC) = XCONE1 + L2 * COS(PSI2)
      DARC      = ARCLEN(JARC) / (JARC - 1)
      DTHETA    = DARC / 2.0
      THETA     = 0.0
        DO 1 J    = 2, JARC-1
        ARCLEN(J) = ARCLEN(J - 1) + DARC
        IF (XARC(J-1) .LT. 0.0) THEN
          THETA     = THETA + DTHETA
          XARC(J)   =  - 2.0 * COS(THETA)
        ELSE IF (XARC(J-1) .LT. 3.0) THEN
          XARC(J)   = XARC(J - 1) + DARC
        ELSE IF (XARC(J-1) .LT. XCONE1) THEN
          XARC(J)   = XARC(J - 1) + DARC * COS(PSI1)
        ELSE
          XARC(J)   = XARC(J - 1) + DARC * COS(PSI2)
        END IF
    1   CONTINUE
      RETURN
      END
```

Appendix A-2: Grid Generating Program *DEHNEN*

```
      PROGRAM     DEMNEN
      PARAMETER   (NMAX = 1000)
      DIMENSION   XB(NMAX)
      CHARACTER   FILE * 40
C
      PRINT *, ' NUMBER OF POINTS?'
      READ *, N
      PRINT *, 'SIZE OF SMALLEST STEP?'
      READ *, DX
      PRINT *, ' TOTAL SPAN?'
      READ *, SPAN
      IF (SPAN .LE. 0.0) SPAN = 1.0
      N1          = N - 1
      DXM1        = 1.0 / FLOAT(N1)
      CALL NRX (DX, SPAN, DXM1, CK)
      PRINT *, ' FINAL CK = ', CK
      EKM1        = EXP(CK) - 1.0
      PRINT *, ' SHALL THE SMALLEST INCREMENT BE FIRST (1)'
      PRINT *, ' OR LAST (2)?'
      READ *, IGO
      IF (IGO .GT. 1) THEN
          DO 1 I      = 1, N
          J           = 1 + N - I
          B           = FLOAT (I - 1) * DXM1
          XB(J)       = SPAN * (1.0 - (EXP(CK * B) - 1.0) / EKM1)
    1     CONTINUE
      ELSE
          DO 2 I      = 1, N
          B           = FLOAT(I - 1) * DXM1
          XB(I)       = SPAN * (EXP(CK * B) - 1.0) / EKM1
    2     CONTINUE
      END IF
      PRINT *, ' NAME OF OUTPUT FILE?'
      READ (*,  100) FILE
      OPEN (7, FILE = FILE, STATUS = 'UNKNOWN')
      REWIND 7
        DO 3 I    = 1, N
        WRITE (7,  101) XB (I)
    3   CONTINUE
      STOP
C
  100 FORMAT  ( A )
  101 FORMAT  ( 2X, E14.7 )
C
      END
      SUBROUTINE  NRX (D, EL, ZETA, CK)
C
C        SOLVE EXPONENTIAL GRID-STRETCHING EQUATION
C        BY NEWTON-RAPHSON ITERATION
C
      CK          = 1.0
        DO 1 K    = 1, 10
        EK        = EXP(CK)
        EKZ       = EXP(CK * ZETA)
        FK        = EL * (EKZ - 1.0) / (EK - 1.0) - D
        FKP       = EL * (ZETA * EKZ - (EKZ - 1.0) * EK / (EK - 1.0)) /
     1    (EK - 1.0)
        IF (ABS(FK/FKP) .LE. 1.0E-5) GO TO 2
        CK        = CK - FK / FKP
    1   CONTINUE
      WRITE (6, *) ' NRX FAILED.'
    2 RETURN
      END
```

Appendix A-3: Grid Spacing Table *EJECVRT*

0.0000000E+00	.2293029E+00
.2272727E-02	.2354554E+00
.4603636E-02	.2425577E+00
.6994545E-02	.2507086E+00
.9446818E-02	.2594168E+00
.1196227E-01	.2683482E+00
.1454182E-01	.2775091E+00
.1718818E-01	.2869050E+00
.1990182E-01	.2965423E+00
.2268545E-01	.3064268E+00
.2554045E-01	.3165655E+00
.2846864E-01	.3269636E+00
.3147227E-01	.3376291E+00
.3455273E-01	.3485686E+00
.3771273E-01	.3597886E+00
.4095318E-01	.3712964E+00
.4427727E-01	.3831000E+00
.4768636E-01	.3952064E+00
.5118318E-01	.4076232E+00
.5477000E-01	.4203591E+00
.5844818E-01	.4334218E+00
.6222136E-01	.4468200E+00
.6609136E-01	.4605618E+00
.7006045E-01	.4746568E+00
.7413136E-01	.4891132E+00
.7830727E-01	.5039409E+00
.8259000E-01	.5191491E+00
.8698273E-01	.5347477E+00
.9148818E-01	.5507466E+00
.9610909E-01	.5671562E+00
.1008491E+00	.5839871E+00
.1057105E+00	.6012500E+00
.1106964E+00	.6189560E+00
.1158105E+00	.6371164E+00
.1210559E+00	.6557430E+00
.1264359E+00	.6748478E+00
.1319541E+00	.6944430E+00
.1376141E+00	.7145410E+00
.1434191E+00	.7351551E+00
.1493732E+00	.7562982E+00
.1554800E+00	.7771149E+00
.1617436E+00	.7972787E+00
.1681682E+00	.8167897E+00
.1741442E+00	.8356479E+00
.1792891E+00	.8538532E+00
.1837018E+00	.8714057E+00
.1874812E+00	.8883053E+00
.1907263E+00	.9045521E+00
.1935359E+00	.9201460E+00
.1960091E+00	.9350871E+00
.1982448E+00	.9493754E+00
.2003418E+00	.9630108E+00
.2023992E+00	.9759934E+00
.2045159E+00	.9883231E+00
.2067908E+00	.1000000E+01
.2093228E+00	
.2122109E+00	
.2155540E+00	
.2194511E+00	
.2240011E+00	

Appendix A-4: Grid Spacing Table *EJECHRZ*

.1000000E-04
.2000000E-02
.4041153E-02
.6124307E-02
.8250325E-02
.1042009E-01
.1263450E-01
.1489448E-01
.1720095E-01
.1955489E-01
.2195727E-01
.2440907E-01
.2691133E-01
.2946507E-01
.3207137E-01
.3473129E-01
.3744594E-01
.4021645E-01
.4304397E-01
.4592967E-01
.4887475E-01
.5188043E-01
.5494796E-01
.5807860E-01
.6127366E-01
.6453447E-01
.6786237E-01
.7125875E-01
.7472502E-01
.7826261E-01
.8187299E-01
.8555766E-01
.8931815E-01
.9315602E-01
.9707286E-01
.1010703E+00
.1051500E+00
.1093136E+00
.1135629E+00
.1178997E+00
.1223256E+00
.1268427E+00
.1314527E+00
.1355267E+00
.1386259E+00
.1408957E+00
.1424818E+00
.1435298E+00
.1441852E+00
.1445936E+00
.1449006E+00
.1452517E+00
.1457925E+00
.1466687E+00
.1480258E+00
.1500093E+00
.1527648E+00
.1564380E+00
.1611744E+00
.1664909E+00
.1719167E+00
.1774542E+00
.1831056E+00
.1888733E+00
.1947596E+00
.2007671E+00
.2068983E+00
.2131555E+00
.2195416E+00
.2260590E+00
.2327105E+00
.2394989E+00
.2464270E+00
.2534976E+00
.2607138E+00
.2680784E+00
.2755945E+00
.2832653E+00
.2910940E+00
.2990837E+00
.3072379E+00
.3155598E+00
.3240530E+00
.3327209E+00
.3415672E+00
.3505955E+00
.3598095E+00
.3692132E+00
.3788104E+00
.3886050E+00
.3986012E+00
.4088031E+00
.4192149E+00
.4298409E+00
.4406856E+00
.4517534E+00
.4630490E+00
.4745770E+00
.4863422E+00
.4983495E+00
.5106039E+00
.5231104E+00
.5358742E+00
.5489007E+00
.5621952E+00
.5757633E+00
.5896106E+00
.6037428E+00
.6181658E+00
.6328856E+00
.6479083E+00
.6632400E+00
.6788873E+00
.6948565E+00
.7111543E+00
.7277875E+00
.7447629E+00
.7620876E+00
.7797688E+00
.7978139E+00
.8162302E+00
.8350255E+00
.8542075E+00
.8737842E+00
.8937638E+00
.9141544E+00
.9349647E+00
.9562031E+00
.9778785E+00
.1000000E+01

Appendix B: Flow Solving Program *JSIAXK*

```
                                                                XKKO
      PROGRAM     JSIAXK
      PARAMETER   (MK = 130, MJ = 115)
      COMMON /  DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1   CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
     2   JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
      COMMON  /CGAS/  CVTH, CVTAH, GAMMP, GAMMR, GAMMT, RUINF, REINF
      COMMON  /DEP/   RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1   MJ)
      COMMON  /DEPP/  RHOP(MK, MJ), RHOUP(MK, MJ), RHOVP(MK, MJ),
     1   RHOEP(MK, MJ)
      COMMON  /DOF/   X(MK, MJ), Y(MK, MJ), RY(MK, MJ)
      COMMON  /ERR/   KFE, JFE
      COMMON  /IBC/   PTIN, Q2CP, TTIN
      COMMON  /NELSON/  RDET, RDZT, QDET, RMU, SL(MK, MJ)
      COMMON  /NOZ/   KNOZ, JNOZIN, JNOZOUT, JNMIN, JNMOUT, JNERIN,
     1   JNEROUT, PINOZ, TINOZ, ALPHA, EFFNOZ
      COMMON  /OLD/   ROLD(MK, MJ), RUOLD(MK, MJ), RVOLD(MK, MJ),
     1   REOLD(MK, MJ), XNORM
      COMMON  /PLT/   RL, RH, RCV, PINF, KSTEP
      COMMON  /TV/    ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
      COMMON  /TVIS/  EP(MK, MJ), CPQPR, CPQPRT
C
      LOGICAL     LAMIN, OOPS
C
      NAMELIST  /JIX/  ALPHA, BETA, CFL, CX, CY, ISMTHX, ISMTHY,
     1   ISTART, JL, KL, KSTEP, LAMIN, NEND, PTIN, REY, RL, TINF, TTIN,
     2   TH, XM
C
C     CALL DROPFILE (0)
C
C        CONSTANTS AND PARAMETERS FOR AIR
C
      CV          = 4290.0
      RC          = 1718.0
      GAMMA       = 1.4
      GAMM1       = GAMMA - 1.0
      GAMMP       = GAMM1 / GAMMA
      GAMMR       = GAMMA * RC
      GAMMT       = GAMMA / GAMM1
      PR          = 0.73
      PRT         = 0.90
      RCV         = 1.0 / CV
      Q2CP        = 0.5 / (GAMMA * CV)
      CPQPR       = GAMMA * CV / PR
      CPQPRT      = GAMMA * CV / PRT
C
C        READ IN THE INPUT DATA
C
      OPEN (5, FILE = 'JIX.DAT', FORM = 'FORMATTED', STATUS = 'OLD')
C     CALL DDIOPON ('LABELS')
      READ (5, JIX)
C     CALL DDIOPOFF ('LABELS')
      CLOSE (5)
C
C        GENERATE THE EXIT PLANE (REFERENCE) INFORMATION
C
      CINF        = SQRT(GAMMA * RC * TINF)
      UINF        = CINF * XM
      RMUINF      = 3.719 E-07
      PINF        = RHOINF * RC * TINF
      CVTH        = CV * TH
      CVTAH       = CV * TINF * (1.0 + 0.5 * GAMM1 * SQRT(PR) * XM ** 2)
      RUINF       = RHOINF * UINF
      REINF       = RHOINF * (CV * TINF + 0.5 * UINF ** 2)
      TCH         = RL / UINF
C
      JLM         = JL - 1
      KLM         = KL - 1
      DETA        = 1.0 / JLM
      DZETA       = 1.0 / KLM
C
C        WRITE OUT THE INPUT AND FLOW FIELD INFORMATION
C
      OPEN (6, FILE = 'JIX.OUT', FORM = 'FORMATTED', STATUS = 'NEW')
      WRITE (6,  100) XM, REY, TH, RHOINF
      WRITE (6,  101) UINF, CINF, TINF, PINF, RMUINF
      WRITE (6,  102) DETA, DZETA, RL
      WRITE (6,  103) CFL, CX, CY, BETA
      WRITE (6,  104) NEND, JL, KL, LAMIN, TTIN, PTIN
      IF (BETA .NE. 0.0) WRITE (6,  105)
C
      NI          = 1
      TMS1        = 0.0
      TMS2        = 0.0
C
      IF (ISTART .EQ. 0) THEN
C
C        READ IN THE GRID POINTS IN CARTESIAN FRAME AND
C        INITIALIZE ALL DEPENDENT VARIABLES
C
        OPEN (1, FILE = 'EJECO', FORM = 'UNFORMATTED', STATUS =
     1     'OLD')
        READ (1)  ((X(K, J), K = 1, KL), J = 1, JL),
     1            ((Y(K, J), K = 1, KL), J = 1, JL)
        READ (1)  ((RHO(K, J), K = 1, KL), J = 1, JL),
     1            ((RHOU(K, J), K = 1, KL), J = 1, JL),
     2            ((RHOV(K, J), K = 1, KL), J = 1, JL),
     3            ((RHOE(K, J), K = 1, KL), J = 1, JL)
        CLOSE (1)
      ELSE
C
C        READ IN THE RESTART DATA FROM PREVIOUS RUNS
C
        OPEN (1, FILE = 'JIXPRV', FORM = 'UNFORMATTED', STATUS =
     1     'OLD')
        READ (1) NI, TMS1, TMS2
        READ (1)  ((RHO(K, J), K = 1, KL), J = 1, JL),
     1            ((RHOU(K, J), K = 1, KL), J = 1, JL),
     2            ((RHOV(K, J), K = 1, KL), J = 1, JL),
     3            ((RHOE(K, J), K = 1, KL), J = 1, JL),
     4            ((EP(K, J), K = 1, KL), J = 1, JL)
        READ (1)  ((X(K, J), K = 1, KL), J = 1, JL),
     1            ((Y(K, J), K = 1, KL), J = 1, JL)
        CLOSE (1)
      END IF
C
C        SOME CONSTANTS FOR EDDY
C
      RDET        = 0.5 / DETA
      RDZT        = 0.5 / DZETA
```

```
      QDET      = 1.0 / DETA
      IF (TM .GT. 0.0) THEN
        TREF    = TM
      ELSE
        TREF    = CVTAM / CV
      END IF
      RMU       = 2.270E - 08 * SQRT(TRE ** 3) / (TREF + 198.6)
C
C       SET NOZZLE PARAMETERS
C
      TTNOZ     = 518.7
      PRATIO    = 1.14
      PTNOZ     = PRATIO * PTIN
      EFFNOZ    = 1.00
      JNOZIN    = 57
      JNOZOUT   = 69
      KNOZ      = 46
      JNMIN     = JNOZIN - 3
      JNMOUT    = JNOZOUT + 3
      JNERIN    = JNMIN - 1
      JNEROUT   = JNMOUT + 1
      DO 1 J = 1, JL
        IF (J.LT.JNMIN) THEN
          KSTART(J) = 1
        ELSE IF (J.LT.JNMOUT) THEN
          KSTART(J) = KNOZ
        ELSE
          KSTART(J) = 1
        ENDIF
    1 CONTINUE
C
      CALL PREAMB
      CALL TRANS
      CALL TMSTEP
C
C       THE FAMOUS "MAIN LOOP."
C
      DO 3 N    = NI, NEND
C
      IF ((N/10)*10 .NE. N) GO TO 2
C
C       DETERMINE THE ALLOWABLE TIME STEP SIZE (DT CFL)
C
      IF ( .NOT. LAMIN) CALL EDDY
      CALL TMSTEP
C
C       CONTROL THE DATA FLOW AND DIFFERENCE OPERATORS
C
    2 CALL PAGE (OOPS)
      IF (OOPS) THEN
        WRITE (6,  106) KFE, JFE, N
        WRITE (*,  106) KFE, JFE, N
        GO TO 4
      END IF
C
      CALL L2NORM
      TMS2      = TMS1 + 10.0 * DTL(2)
      T         = TMS2 / TCH
      IF (MOD(N,100) .EQ. 0) THEN
        WRITE (6,  107) N, DTL (2), XNORM, T, DTL (JLM)
        WRITE (*,  108) N, XNORM, SECS
      END IF
      TMS1      = TMS2
C
C       END OF YOU KNOW WHAT
C
    3 CONTINUE
C
C       WRITE THE RESTART DATA INTO RESTART TAPE
C
      NEND      = NEND + 1
      OPEN (2, FILE = 'JIXRGO', FORM = 'UNFORMATTED', STATUS = 'NEW')
      WRITE (2) NEND, TMS1, TMS2
      WRITE (2) ((RHO(K, J), K = 1, KL), J = 1, JL),
     1          ((RHOU(K, J), K = 1, KL), J = 1, JL),
     2          ((RHOV(K, J), K = 1, KL), J = 1, JL),
     3          ((RHOE(K, J), K = 1, KL), J = 1, JL),
     4          ((EP(K, J), K = 1, KL), J = 1, JL)
      WRITE (2) ((X(K, J), K = 1, KL), J = 1, JL),
     1          ((Y(K, J), K = 1, KL), J = 1, JL)
      CLOSE (2)
C
    4 CALL PLOTCN
C
      STOP
C
  100 FORMAT ( '1     DR. SHANG''S FAMOUS EXPLICIT CODE ADAPTED FOR ',
     1   'AXIALLY SYMMETRIC INTERNAL FLOW' // 5X, 'REFERENCE MACH NUMB',
     2   'ER    =', F10.4, / 5X, 'REYNOLDS NUMBER', T31, '=', E15.7,
     3   T66, 'AA', T80, 'AA', T105, 'AA' / 5X, 'SURFACE TEMPERATURE (',
     4   'R) =', F10.4, T66, 'VV', T80, 'VV', T105, 'VV' / 5X,
     5   'RHOREF', T31, '=', E15.7, / T66, 'JJ', 3X,
     6   'ASSSSA   II   AAAAA AA   VA    AV   II' )
  101 FORMAT ( 5X, 'UREF', T31, '=', F10.4, T66, 'JJ  ASV  VSA  II',
     1   '   AAV', T90, 'VAA   VA   AV   II' / 5X, 'CREF', T31, '=',
     2   F10.4, T66, 'JJ  SS', T80, 'II  AA', T91, 'AA    VAAV   II' /
     3   5X, 'TREF', T31, '=', F10.4, T66, 'JJ  VSA', T80, 'II  AA',
     4   T91, 'AA    VXXV    II' / 5X, 'PREF', T31, '=', F10.4, T66,
     5   'JJ  VSSSSA   II  AA', 5X, 'AA', 5X, 'XX', 5X, 'II' / 5X,
     6   'MUREF', T31, '=', E15.7, T66, 'JJ', 7X, 'VSA  II  AA', T91,
     7   'AA', 5X, 'XX', 5X, 'II' / T66, 'JJ', 8X, 'SS  II  AA', T91,
     8   'AA    AXXA    II' )
  102 FORMAT ( 5X, 'DETA', T31, '=', F10.4, T66, 'JJ', 8X, 'SS  II',
     1   '  AA', 5X, 'AA   AV  VA   II' / 5X, 'DZETA', T31, '=', F10.4,
     2   T66, 'JJ  VSA  ASV  II  VAA   AAA   AV  VA   II' / 5X,
     3   'REFERENCE LENGTH', T31, '=', F10.4, T66, 'JJ   VSSSSV   II',
     4   '   VAAAA AA   AV   VA   II' / T66, 'JJ' )
  103 FORMAT ( 5X, 'CFL', T31, '=', E15.7, T66, 'JJ' / 5X, 'CX', T31,
     1   '=', F10.4, T66, 'JJ' / 5X, 'CY', T31, '=', F10.4, T66, 'JJ' /
     2   5X, 'BETA', T31, '=', F10.4, T66, 'JJ' / T66, 'JJ' )
  104 FORMAT ( 5X, 'NEND', T31, '=', I8, T60, 'VJA  AJV' / 5X, 'JL',
     1   T31, '=', 2X, I6, T61, 'VJJJJV' / 5X, 'KL', T31, '=', 2X, I6 /
     2   5X, 'LAMIN', T31, '=', 7X, L1 , 5X, 'TTIN', T31, '=', F10.4 /
     3   5X, 'PTIN', T31, '=', F11.4 / )
  105 FORMAT ( 12X, 'LOCAL CFL TIME STEP CALCULATION (''IME MARP)' )
  106 FORMAT ( '0 MEANINGLESS TEMPERATURE AT K =', I4, ', J =', I4,
     1   ', N =', I7 )
  107 FORMAT ( ' N =', I7, ' DTCFL =', E15.7, ' L2NORM =', E15.7,
     1   ' TMS2/TCH =', E15.7, ' DTMAX =', E15.7 )
  108 FORMAT ( ' N =', I7, ' L2NORM =', G12.5, ' CP TIME =', G12.5 )
C
```

```
      END
      SUBROUTINE  BC
      PARAMETER   (MK = 130, MJ = 115)
      COMMON      DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1  CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DI, CX, CY, L, JL, KL,
     2  JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
      COMMON  /CGAS/   CVTH, CVTAH, GAMMP, GAMMR, GAMNT, RUINF, REINF
      COMMON  /DEP/   RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1  MJ)
      COMMON  /DEPP/   RHOP(MK, MJ), RHOUP(MK, MJ), RHOVP(MK, MJ),
     1  RHOEP(MK, MJ)
      COMMON  /DOF/   X(MK, MJ), Y(MK, MJ), RY(MK, MJ)
      COMMON  /IBC/   PTIN, Q2CP, TTIN
      COMMON  /NOZ/   KNOZ, JNOZIN, JNOZOUT, JNMIN, JNMOUT, JNERIN,
     1  JNEROUT, PTNOZ, TTNOZ, ALPHA, EFFNOZ
      COMMON  /PLT/   RL, RH, RCV, PINF, KSTEP
      COMMON  /TV/   ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
C
      IF (L .EQ. 1) THEN
C
C
C     PREDICTOR SWEEP
C
C     UPSTREAM BOUNDARY CONDITIONS.
C
        DO 1 J    = 2, JLM
        JP = J + 1
        JM = J - 1
        RHOP(1, J) = PTIN / (RC * TTIN)
        RHOUP(1,J) = RHOUP(2,J)
        RHOVP(1,J) = RHOVP(2,J) * Y(2,J) / Y(1,J)
        U = RHOUP(1,J) / RHOP(1,J)
        V = RHOVP(1,J) / RHOP(1,J)
        PSI = ATAN((X(1,JP)-X(1,JM))/(Y(1,JP)-Y(1,JM)))
        UPRIME = RHOUP(1,J)*COS(PSI)-RHOVP(1,J)*SIN(PSI)
        T = TTIN - Q2CP * (U ** 2 + V ** 2)
        RHOEP(1, J) = RHOP(1, J) * (CV * T + 0.5 * (U ** 2 +
     1    V ** 2))
C
C           IF OUTFLOW, TREAT LIKE DOWNSTREAM B.C.
C
        IF (UPRIME .LE. 0.0) THEN
        E2 = RHOEP(2, J) / RHOP(2, J)
        U2 = RHOUP(2, J) / RHOP(2, J)
        V2 = RHOVP(2, J) / RHOP(2, J)
        T = (E2 - 0.5 * (U2 ** 2 + V2 ** 2)) / CV
        RHOP(1, J) = PTIN / (T * RC)
        U = RHOUP(1, J) / RHOP(1, J)
        V = RHOVP(1, J) / RHOP(1, J)
        E = CV * T + 0.5 * (U ** 2 + V ** 2)
        RHOEP(1, J) = E * RHOP(1, J)
        ENDIF
C
C
C     DOWNSTREAM BOUNDARY CONDITIONS.
C
        EM = RHOEP(KLM, J) / RHOP(KLM, J)
        UM = RHOUP(KLM, J) / RHOP(KLM, J)
        VM = RHOVP(KLM, J) / RHOP(KLM, J)
        T = (EM - 0.5 * (UM ** 2 + VM ** 2)) / CV
        RHOP(KL, J) = PTIN / (T * RC)
        U = RHOUP(KL, J) / RHOP(KL, J)
        V = RHOVP(KL, J) / RHOP(KL, J)
        E = CV * T + 0.5 * (U ** 2 + V ** 2)
        RHOEP(KL, J) = E * RHOP(KL, J)
    1   CONTINUE
C
C     ON THE SOLID CONTOUR, NO SLIP FOR U, V.
C     CONSTANT SURFACE TEMP. AND DP / DN = 0.
C
        DO 2 K    = 1, KL
        RH        = 1.0 / RHOP(K, JLM)
        RHOUP(K, JL) = 0.0
        RHOVP(K, JL) = 0.0
        RHOEP(K, JL) = RHOEP(K, JLM) - 0.5 * RHOP(K, JLM) *
     1    ((RHOUP(K, JLM) * RH) ** 2 + (RHOVP(K, JLM) * RH) ** 2)
        RHOP(K, JL) = RHOEP(K, JL) / CVTH
C
C     ON THE AXIS, ZERO RADIAL GRADIENT.
C
        RHOVP(K, 1) = 0.0
        RHOUP(K, 1) = RHOUP(K, 2)
        RHOEP(K,1)= RHOEP(K,2) - 0.5 * RHOVP(K,2)**2 / RHOP(K,2)
        RHOP(K, 1) = RHOP(K, 2)
    2   CONTINUE
C
C     PRIMARY NOZZLE BOUNDARY CONDITIONS
C
        KNP = KNOZ + 1
        RHNEARI = 1.0 / RHOP(KNOZ,JNERIN)
        UNEARI = RHOUP(KNOZ,JNERIN) * RHNEARI
        VNEARI = RHOVP(KNOZ,JNERIN) * RHNEARI
        TNEARI = (RHOEP(KNOZ,JNERIN) * RHNEARI - 0.5 *
     1    (UNEARI ** 2 + VNEARI ** 2)) * RCV
        PNEARI = RHOP(KNOZ,JNERIN) * RC * TNEARI
        RHNEARO = 1.0 / RHOP(KNOZ,JNEROUT)
        UNEARO = RHOUP(KNOZ,JNEROUT) * RHNEARO
        VNEARO = RHOVP(KNOZ,JNEROUT) * RHNEARO
        TNEARO = (RHOEP(KNOZ,JNEROUT) * RHNEARO - 0.5 *
     1    (UNEARO ** 2 + VNEARO ** 2)) * RCV
        PNEARO = RHOP(KNOZ,JNEROUT) * RC * TNEARO
C
C     SET NOZZLE EXIT PRESSURE TO AVERAGE LOCAL PRESSURE
C     AND CALCULATE MEAN EXIT VELOCITY
C
C       PNOZ = 0.5 * (PNEARI + PNEARO)
C         IF (PNOZ .GE. (0.95 * PTNOZ)) THEN
C         PNOZ = 0.95 * PTNOZ
C         ENDIF
C       TNOZI = (PNOZ/PTNOZ)**GAMMP * TTNOZ
C       TNOZI = 507.42
C       CNOZ = SQRT(2.0/GAMM1*(TTNOZ/TNOZI-1.0))
C       ANOZ = SQRT(GAMMR * TNOZI)
C       VELNOZ = EFFNOZ ** 0.5 * CNOZ * ANOZ
C       TNOZ = TTNOZ - EFFNOZ * (TTNOZ - TNOZI)
C
C     GIVE EXIT VELOCITY A PARABOLIC PROFILE
C     AND PRESSURE A LINEAR PROFILE
C
C       ZJMAX = SQRT ((X(KNOZ, JNOZOUT) - X(KNOZ, JNOZIN))
C    1    ** 2 + (Y(KNOZ, JNOZOUT) - Y(KNOZ, JNOZIN)) ** 2)
```

```
      DO 3     J = JNOZIN + 1, JNOZOUT - 1
C     ZJ = SQRT ( X(KNOZ, J) - X(KNOZ, JNOZIN))
C    1   ** 2 +  (KNOZ, J) - Y(KNOZ, JNOZIN)) ** 2)
C     Z = ZJ / ZMAX
C     VELNOZJ = 6.0 * VELNOZ * Z * (1.0 - Z)
C     PNOZJ = PNEARI + Z * (PNEARO - PNEARI)
C     TNOZI = (PNOZJ/PTNOZ)**GAMMP * TTNOZ
C     RHONOZ = PTNOZ / (RC * TTNOZ)
C     RHOUP(KNOZ,J) = ( RHOUP(KNP,J) * ZTX(KNP,J) +
C    1    RHOVP(KNP,J) * ZTY(KNP,J) * Y(KNP,J) / Y(KNOZ,J)
C    2    ) / ( ZTX(KNOZ,J) + TAN(ALPHA) * ZTY(KNOZ,J) )
C     RHOVP(KNOZ,J) = RHOUP(KNOZ,J) * TAN(ALPHA)
C     RHOUP(KNOZ,J) = RHOUP(KNP,J)
C     RHOVP(KNOZ,J) = RHOVP(KNP,J) * Y(KNP,J) / Y(KNOZ,J)
C     RHOVEL = SQRT(RHOUP(KNOZ,J)**2 + RHOVP(KNOZ,J)**2)
C     RHOUP(KNOZ,J) = RHOVEL * COS(ALPHA)
C     RHOVP(KNOZ,J) = RHOVEL * SIN(ALPHA)
C     U = RHOUP(KNOZ,J) / RHOP(KNP,J)
C     V = RHOVP(KNOZ,J) / RHOP(KNP,J)
C     TNOZI = TTNOZ - Q2CP * (U ** 2 + V ** 2)
C     TNOZ = TTNOZ - EFFNOZ * (TTNOZ - TNOZI)
C     RGAMM1 = 1.0 / GAMM1
C     RHOP(KNOZ,J) = RHONOZ * (TNOZ / TTNOZ) ** RGAMM1
C     U = RHOUP(KNOZ,J) / RHOP(KNOZ,J)
C     V = RHOVP(KNOZ,J) / RHOP(KNOZ,J)
C     UNOZ = VELNOZJ * COS (ALPHA)
C     VNOZ = VELNOZJ * SIN (ALPHA)
C     RHOUP(KNOZ,J) = RHOP(KNOZ,J) * UNOZ
C     RHOVP(KNOZ,J) = RHOP(KNOZ,J) * VNOZ
C     RHOEP(KNOZ,J) = RHOP(KNOZ,J) * (CV * TNOZ
C    1    + 0.5 * VELNOZJ ** 2)
      E = CV * TNOZ + 0.5 * (U ** 2 + V ** 2)
      RHOEP(KNOZ,J) = E * RHOP(KNOZ,J)
    3 CONTINUE
C
C     NO SLIP / NO PENETRATION ON NOZZLE SIDES
C
      DO 4     K = 1, KNOZ - 1
C
C     OUTSIDE INNER WALL
C
      RH = 1.0 / RHOP(K,JNERIN)
      RHOUP(K,JNWIN) = 0.0
      RHOVP(K,JNWIN) = 0.0
      RHOEP(K,JNWIN) = RHOEP(K,JNERIN) - 0.5 *
     1  RHOP(K,JNERIN) * ((RHOUP(K,JNERIN) * RH)**2
     2  + (RHOVP(K,JNERIN) * RH)**2)
      RHOP(K,JNWIN) = RHOEP(K,JNWIN) / CVTH
C
C     OUTSIDE OUTER WALL
C
      RH = 1.0 / RHOP(K,JNEROUT)
      RHOUP(K,JNWOUT) = 0.0
      RHOVP(K,JNWOUT) = 0.0
      RHOEP(K,JNWOUT) = RHOEP(K,JNEROUT) - 0.5 *
     1  RHOP(K,JNEROUT) * ((RHOUP(K,JNEROUT) * RH)**2
     2  + (RHOVP(K,JNEROUT) * RH)**2)
      RHOP(K,JNWOUT) = RHOEP(K,JNWOUT) / CVTH
C
C     IGNORES ALL POINTS BLOCKED BY NOZZLE STRUCTURE
C
      JCL = (JNOZIN + JNOZOUT) * 0.5
      DO 4 J = JNWIN + 1, JNWOUT - 1
      RHOUP(K,J) = 0.0
      RHOVP(K,J) = 0.0
        IF (J.LE.JCL) THEN
        RHOEP(K,J) = RHOEP(K, JNWIN)
        RHOP(K,J) = RHOP(K, JNWIN)
        ELSE
        RHOEP(K,J) = RHOEP(K, JNWOUT)
        RHOP(K,J) = RHOP(K, JNWOUT)
        ENDIF
    4 CONTINUE
      DO 5 J = JNWIN + 1, JNOZIN - 1
C
C     INNER WALL END
C
      KNER = KNOZ + 1
      RH = 1.0 / RHOP(KNER,J)
      RHOUP(KNOZ,J) = 0.0
      RHOVP(KNOZ,J) = 0.0
      RHOEP(KNOZ,J) = RHOEP(KNER,J) - 0.5 *
     1  RHOP(KNER,J) * ((RHOUP(KNER,J) * RH)**2
     2  + (RHOVP(KNER,J) * RH)**2)
      RHOP(KNOZ,J) = RHOEP(KNOZ,J) / CVTH
    5 CONTINUE
      DO 6 J = JNOZOUT + 1, JNWOUT - 1
C
C     OUTER WALL END
C
      KNER = KNOZ + 1
      RH = 1.0 / RHOP(KNER,J)
      RHOUP(KNOZ,J) = 0.0
      RHOVP(KNOZ,J) = 0.0
      RHOEP(KNOZ,J) = RHOEP(KNER,J) - 0.5 *
     1  RHOP(KNER,J) * ((RHOUP(KNER,J) * RH)**2
     2  + (RHOVP(KNER,J) * RH)**2)
      RHOP(KNOZ,J) = RHOEP(KNOZ,J) / CVTH
    6 CONTINUE
C
C     OUTSIDE INNER WALL END
C
      JC = JNWIN
      JNER = JC - 1
      KC = KNOZ
      KNER = KC + 1
      RHKNER = 1.0 / RHOP(KNER,JC)
      RHJNER = 1.0 / RHOP(KC,JNER)
      RHOUP(KC,JC) = 0.0
      RHOVP(KC,JC) = 0.0
      RHOEP(KC,JC) = 0.5 * RHOEP(KC,JNER) + 0.5
     1  * RHOEP(KNER,JC) - 0.25 * RHOP(KC,JNER) *
     2  ((RHOUP(KC,JNER)*RHJNER)**2 + (RHOVP(KC,JNER)
     3  *RHJNER)**2) - 0.25 * RHOP(KNER,JC) *
     4  ((RHOUP(KNER,JC)*RHKNER)**2 + (RHOVP(KNER,JC)
     5  *RHKNER)**2)
      RHOP(KC,JC) = RHOEP(KC,JC) / CVTH
C
C     INSIDE OUTER WALL END
C
      JC = JNOZOUT
      JNER = JC - 1
```

```
      KC = KNOZ
      KNER = KC + 1
      RHKNER = 1.0 / RHOP(KNER,JC)
      RHJNER = 1.0 / RHOP(KC,JNER)
      RHOUP(KC,JC) = 0.0
      RHOVP(KC,JC) = 0.0
      RHOEP(KC,JC) = 0.5 * RHOEP(KC,JNER) + 0.5
     1  * RHOEP(KNER,JC) - 0.25 * RHOP(KC,JNER) *
     2  ((RHOUP(KC,JNER)*RHJNER)**2 + (RHOVP(KC,JNER)
     3  *RHJNER)**2) - 0.25 * RHOP(KNER,JC) *
     4  ((RHOUP(KNER,JC)*RHKNER)**2 + (RHOVP(KNER,JC)
     5  *RHKNER)**2)
      RHOP(KC,JC) = RHOEP(KC,JC) / CVTH
C
CC    OUTSIDE OUTER WALL END
C
      JC = JNMOUT
      JNER = JC + 1
      KC = KNOZ
      KNER = KC + 1
      RHKNER = 1.0 / RHOP(KNER,JC)
      RHJNER = 1.0 / RHOP(KC,JNER)
      RHOUP(KC,JC) = 0.0
      RHOVP(KC,JC) = 0.0
      RHOEP(KC,JC) = 0.5 * RHOEP(KC,JNER) + 0.5
     1  * RHOEP(KNER,JC) - 0.25 * RHOP(KC,JNER) *
     2  ((RHOUP(KC,JNER)*RHJNER)**2 + (RHOVP(KC,JNER)
     3  *RHJNER)**2) - 0.25 * RHOP(KNER,JC) *
     4  ((RHOUP(KNER,JC)*RHKNER)**2 + (RHOVP(KNER,JC)
     5  *RHKNER)**2)
      RHOP(KC,JC) = RHOEP(KC,JC) / CVTH
C
CC    INSIDE INNER WALL END
C
      JC = JNOZIN
      JNER = JC + 1
      KC = KNOZ
      KNER = KC + 1
      RHKNER = 1.0 / RHOP(KNER,JC)
      RHJNER = 1.0 / RHOP(KC,JNER)
      RHOUP(KC,JC) = 0.0
      RHOVP(KC,JC) = 0.0
      RHOEP(KC,JC) = 0.5 * RHOEP(KC,JNER) + 0.5
     1  * RHOEP(KNER,JC) - 0.25 * RHOP(KC,JNER) *
     2  ((RHOUP(KC,JNER)*RHJNER)**2 + (RHOVP(KC,JNER)
     3  *RHJNER)**2) - 0.25 * RHOP(KNER,JC) *
     4  ((RHOUP(KNER,JC)*RHKNER)**2 + (RHOVP(KNER,JC)
     5  *RHKNER)**2)
      RHOP(KC,JC) = RHOEP(KC,JC) / CVTH
    ELSE
C
CC    CORRECTOR SWEEP
CC
CC    UPSTREAM BOUNDARY CONDITIONS.
C
      DO 7 J    = 2, JLM
      JP = J + 1
      JM = J - 1
      RHO(1, J) = PTIN / (RC * TTIN)
      RHOU(1,J) = RHOU(2,J)
      U = RHOU(1,J) / RHO(1,J)
      V = RHOV(1,J) / RHO(1,J)
      PSI = ATAN((X(1,JP)-X(1,JM))/(Y(1,JP)-Y(1,JM)))
      UPRIME = RHOU(1,J)*COS(PSI)-RHOV(1,J)*SIN(PSI)
      T = TTIN - Q2CP * (U ** 2 + V ** 2)
      RHOE(1, J) = RHO(1, J) * (CV * T + 0.5 * (U ** 2 +
     1    V ** 2))
C
C
C       IF OUTFLOW, TREAT LIKE DOWNSTREAM B.C.
C
      IF (UPRIME .LE. 0.0) THEN
      E2 = RHOE(2, J) / RHO(2, J)
      U2 = RHOU(2, J) / RHO(2, J)
      V2 = RHOV(2, J) / RHO(2, J)
      T = (E2 - 0.5 * (U2 ** 2 + V2 ** 2)) / CV
      RHO(1, J) = PTIN / (T * RC)
      U = RHOU(1, J) / RHO(1, J)
      V = RHOV(1, J) / RHO(1, J)
      E = CV * T + 0.5 * (U ** 2 + V ** 2)
      RHOE(1, J) = E * RHO(1, J)
      ENDIF
C
C
CC    DOWNSTREAM BOUNDARY CONDITIONS.
C
      EM = RHOE(KLM, J) / RHO(KLM, J)
      UM = RHOU(KLM, J) / RHO(KLM, J)
      VM = RHOV(KLM, J) / RHO(KLM, J)
      T = (EM - 0.5 * (UM ** 2 + VM ** 2)) / CV
      RHOU(KL, J) = RHOU(KLM, J)
      RHOV(KL,J) = RHOV(KLM,J)*Y(KLM,J)/Y(KL,J)
      RHO(KL, J) = PTIN / (T * RC)
      U = RHOU(KL, J) / RHO(KL, J)
      V = RHOV(KL, J) / RHO(KL, J)
      E = CV * T + 0.5 * (U ** 2 + V ** 2)
      RHOE(KL, J) = E * RHO(KL, J)
    7 CONTINUE
C
CC    ON THE SOLID CONTOUR, NO SLIP FOR U, V.
CC    CONSTANT SURFACE TEMP. AND DP / DN = 0.
C
      DO 8 K    = 1, KL
      RH        = 1.0 / RHO(K, JLM)
      RHOU(K, JL) = 0.0
      RHOV(K, JL) = 0.0
      RHOE(K, JL) = RHOE(K, JLM) - 0.5 * RHO(K, JLM) *
     1   ((RHOU(K, JLM) * RH) ** 2 + (RHOV(K, JLM) * RH) ** 2)
      RHO(K, JL) = RHOE(K, JL) / CVTH
C
CC    ON THE AXIS, ZERO RADIAL GRADIENT.
C
      RHOV(K, 1) = 0.0
      RHOU(K, 1) = RHOU(K, 2)
      RHOE(K,1) = RHOE(K,2) - 0.5 * RHOV(K,2)**2 / RHO(K,2)
      RHO(K, 1) = RHO(K, 2)
    8 CONTINUE
C
CC    PRIMARY NOZZLE BOUNDARY CONDITIONS
C
      KNP = KNOZ + 1
      RHNEAR1 = 1.0 / RHO(KNOZ,JNERIN)
```

```
      UNEARI = RHOU(KNOZ,JNERIN) * RHNEARI
      VNEARI = RHOV(KNOZ,JNERIN) * RHNEARI
      TNEARI = (RHOE(KNOZ,JNERIN) * RHNEARI - 0.5 *
     1   (UNEARI ** 2 + VNEARI ** 2)) * RCV
      PNEARI = RHO(KNOZ,JNERIN) * RC * TNEARI
      RHNEARO = 1.0 / RHO(KNOZ,JNEROUT)
      UNEARO = RHOU(KNOZ,JNEROUT) * RHNEARO
      VNEARO = RHOV(KNOZ,JNEROUT) * RHNEARO
      TNEARO = (RHOE(KNOZ,JNEROUT) * RHNEARO - 0.5 *
     1   (UNEARO ** 2 + VNEARO ** 2)) * RCV
      PNEARO = RHO(KNOZ,JNEROUT) * RC * TNEARO
C
C     SET NOZZLE EXIT PRESSURE TO AVERAGE LOCAL PRESSURE
C     AND CALCULATE MEAN EXIT VELOCITY
C
        PNOZ = 0.5 * (PNEARI + PNEARO)
          IF (PNOZ .GE. (0.95 * PTNOZ)) THEN
          PNOZ = 0.95 * PTNOZ
          ENDIF
        TNOZI =(PNOZ/PTNOZ)**GAMMP * TTNOZ
        TNOZI = 507.42
        CNOZ = SQRT(2.0/GAMM1*(TTNOZ/TNOZI-1.0))
        ANOZ = SQRT(GAMMR * TNOZI)
        VELNOZ = EFFNOZ ** 0.5 * CNOZ * ANOZ
        TNOZ = TTNOZ - EFFNOZ * (TTNOZ - TNOZI)
C
C     GIVE EXIT VELOCITY A PARABOLIC PROFILE
C     AND PRESSURE A LINEAR PROFILE
C
        ZJMAX = SQRT (( X(KNOZ, JNOZOUT) - X(KNOZ, JNOZIN))
     1     ** 2 + ( Y(KNOZ, JNOZOUT) - Y(KNOZ, JNOZIN)) ** 2)
        DO 9    J = JNOZIN + 1, JNOZOUT - 1
        ZJ = SQRT (( X(KNOZ, J) - X(KNOZ, JNOZIN))
     1     ** 2 + ( Y(KNOZ, J) - Y(KNOZ, JNOZIN)) ** 2)
        Z = ZJ / ZJMAX
        VELNOZJ = 6.0 * VELNOZ * Z * (1.0 - Z)
        PNOZJ = PNEARI + Z * (PNEARO - PNEARI)
        TNOZI =(PNOZJ/PTNOZ)**GAMMP * TTNOZ
        TNOZ = TTNOZ - EFFNOZ * (TTNOZ - TNOZI)
        UNOZ = VELNOZJ * COS (ALPHA)
        VNOZ = VELNOZJ * SIN (ALPHA)
        RHO(KNOZ,J) = PNOZJ / (RC * TNOZ)
        RHOU(KNOZ,J) = RHO(KNOZ,J) * UNOZ
        RHOV(KNOZ,J) = RHO(KNOZ,J) * VNOZ
        RHOE(KNOZ,J) = RHO(KNOZ,J) * (CV * TNOZ
     1     + 0.5 * VELNOZJ ** 2
        RHONOZ = PTNOZ / (RC * TTNOZ)
        RHOU(KNOZ,J) = ( RHOU(KNP,J) * ZTX(KNP,J) +
     1      RHOV(KNP,J) * ZTY(KNP,J) * Y(KNP,J) / Y(KNOZ,J) )
     2      / ( ZTX(KNOZ,J) + TAN(ALPHA) * ZTY(KNOZ,J) )
        RHOV(KNOZ,J) = RHOU(KNOZ,J) * TAN(ALPHA)
        RHOU(KNOZ,J) = RHOU(KNP,J)
        RHOV(KNOZ,J) = RHOV(KNP,J) * Y(KNP,J) / Y(KNOZ,J)
        RHOVEL = SQRT(RHOU(KNOZ,J)**2 + RHOV(KNOZ,J)**2)
        RHOU(KNOZ,J) = RHOVEL * COS(ALPHA)
        RHOV(KNOZ,J) = RHOVEL * SIN(ALPHA)
        U = RHOU(KNOZ,J) / RHO(KNP,J)
        V = RHOV(KNOZ,J) / RHO(KNP,J)
        TNOZI = TTNOZ - Q2CP * (U ** 2 + V ** 2)
        TNOZ = TTNOZ - EFFNOZ * (TTNOZ - TNOZI)
        RHO(KNOZ,J) = RHONOZ * (TNOZ / TTNOZ) ** RGAMM1
        U = RHOU(KNOZ,J) / RHO(KNOZ,J)
        V = RHOV(KNOZ,J) / RHO(KNOZ,J)
        E = CV * TNOZ + 0.5 * (U ** 2 + V ** 2)
        RHOE(KNOZ,J) = E * RHO(KNOZ,J)
    9   CONTINUE
C
C     NO SLIP / NO PENETRATION ON NOZZLE SIDES
C
      DO 10   K = 1, KNOZ - 1
C
C     OUTSIDE INNER WALL
C
      RH = 1.0 / RHO(K,JNERIN)
      RHOU(K,JNWIN) = 0.0
      RHOV(K,JNWIN) = 0.0
      RHOE(K,JNWIN) = RHOE(K,JNERIN) - 0.5 *
     1   RHO(K,JNERIN) * ((RHOU(K,JNERIN) * RH)**2
     2   + (RHOV(K,JNERIN) * RH)**2)
      RHO(K,JNWIN) = RHOE(K,JNWIN) / CVTW
C
C     OUTSIDE OUTER WALL
C
      RH = 1.0 / RHO(K,JNEROUT)
      RHOU(K,JNWOUT) = 0.0
      RHOV(K,JNWOUT) = 0.0
      RHOE(K,JNOZOUT) = RHOE(K,JNEROUT) - 0.5 *
     1   RHO(K,JNEROUT) * ((RHOU(K,JNEROUT) * RH)**2
     2   + (RHOV(K,JNEROUT) * RH)**2)
      RHO(K,JNWOUT) = RHOEP(K,JNWOUT) / CVTW
C
C     IGNORES ALL POINTS BLOCKED BY NOZZLE STRUCTURE
C
      JCL = (JNOZIN + JNOZOUT) * 0.5
      DO 10    J = JNWIN + 1, JNWOUT - 1
      IF (K.NE.KNOZ) THEN
      RHOU(K,J) = 0.0
      RHOV(K,J) = 0.0
        IF (J.LE.JCL) THEN
        RHOE(K,J) = RHOE(K, JNWIN)
        RHO(K,J) = RHO(K, JNWIN)
        ELSE
        RHOE(K,J) = RHOE(K, JNWOUT)
        RHO(K,J) = RHO(K, JNWOUT)
        ENDIF
      ENDIF
   10 CONTINUE
      DO 11 J = JNWIN + 1, JNOZIN - 1
C
C     INNER WALL END
C
        KNER = KNOZ + 1
        RH = 1.0 / RHO(KNER,J)
        RHOU(KNOZ,J) = 0.0
        RHOV(KNOZ,J) = 0.0
        RHOE(KNOZ,J) = RHOE(KNER,J) - 0.5 *
     1    RHO(KNER,J) * ((RHOU(KNER,J) * RH)**2
     2    + (RHOV(KNER,J) * RH)**2)
        RHO(KNOZ,J) = RHOE(KNOZ,J) / CVTW
   11 CONTINUE
      DO 12 J = JNOZOUT + 1, JNWOUT - 1
```

```
C
C       OUTER WALL END
C
          KNER = KNOZ + 1
          RH = 1.0 / RHO(KNER,J)
          RHOU(KNOZ,J) = 0.0
          RHOV(KNOZ,J) = 0.0
          RHOE(KNOZ,J) = RHOE(KNER,J) - 0.5 *
     1     RHO(KNER,J) * ((RHOU(KNER,J) * RH)**2
     2     + (RHOV(KNER,J) * RH)**2)
          RHO(KNOZ,J) = RHOE(KNOZ,J) / CVTH
   12   CONTINUE
C
C       OUTSIDE INNER WALL END
C
          JC = JNHIN
          JNER = JC - 1
          KC = KNOZ
          KNER = KC + 1
          RHKNER = 1.0 / RHO(KNER,JC)
          RHJNER = 1.0 / RHO(KC,JNER)
          RHOU(KC,JC) = 0.0
          RHOV(KC,JC) = 0.0
          RHOE(KC,JC) = 0.5 * RHOE(KC,JNER) + 0.5
     1     * RHOE(KNER,JC) - 0.25 * RHO(KC,JNER) *
     2     ((RHOU(KC,JNER)*RHJNER)**2 + (RHOV(KC,JNER)
     3     *RHJNER)**2) - 0.25 * RHO(KNER,JC) *
     4     ((RHOU(KNER,JC)*RHKNER)**2 + (RHOV(KNER,JC)
     5     *RHKNER)**2)
          RHO(KC,JC) = RHOE(KC,JC) / CVTH
C
C       INSIDE OUTER WALL END
C
          JC = JNOZOUT
          JNER = JC - 1
          KC = KNOZ
          KNER = KC + 1
          RHKNER = 1.0 / RHO(KNER,JC)
          RHJNER = 1.0 / RHO(KC,JNER)
          RHOU(KC,JC) = 0.0
          RHOV(KC,JC) = 0.0
          RHOE(KC,JC) = 0.5 * RHOE(KC,JNER) + 0.5
     1     * RHOE(KNER,JC) - 0.25 * RHO(KC,JNER) *
     2     ((RHOU(KC,JNER)*RHJNER)**2 + (RHOV(KC,JNER)
     3     *RHJNER)**2) - 0.25 * RHO(KNER,JC) *
     4     ((RHOU(KNER,JC)*RHKNER)**2 + (RHOV(KNER,JC)
     5     *RHKNER)**2)
          RHO(KC,JC) = RHOE(KC,JC) / CVTH
C
C       OUTSIDE OUTER WALL END
C
          JC = JNHOUT
          JNER = JC + 1
          KC = KNOZ
          KNER = KC + 1
          RHKNER = 1.0 / RHO(KNER,JC)
          RHJNER = 1.0 / RHO(KC,JNER)
          RHOU(KC,JC) = 0.0
          RHOV(KC,JC) = 0.0
          RHOE(KC,JC) = 0.5 * RHOE(KC,JNER) + 0.5
```

```
     1     * RHOE(KNER,JC) - 0.25 * RHO(KC,JNER) *
     2     ((RHOU(KC,JNER)*RHJNER)**2 + (RHOV(KC,JNER)
     3     *RHJNER)**2) - 0.25 * RHO(KNER,JC) *
     4     ((RHOU(KNER,JC)*RHKNER)**2 + (RHOV(KNER,JC)
     5     *RHKNER)**2)
          RHO(KC,JC) = RHOE(KC,JC) / CVTH
C
C       INSIDE INNER WALL END
C
          JC = JNOZIN
          JNER = JC + 1
          KC = KNOZ
          KNER = KC + 1
          RHKNER = 1.0 / RHO(KNER,JC)
          RHJNER = 1.0 / RHO(KC,JNER)
          RHOU(KC,JC) = 0.0
          RHOV(KC,JC) = 0.0
          RHOE(KC,JC) = 0.5 * RHOE(KC,JNER) + 0.5
     1     * RHOE(KNER,JC) - 0.25 * RHO(KC,JNER) *
     2     ((RHOU(KC,JNER)*RHJNER)**2 + (RHOV(KC,JNER)
     3     *RHJNER)**2) - 0.25 * RHO(KNER,JC) *
     4     ((RHOU(KNER,JC)*RHKNER)**2 + (RHOV(KNER,JC)
     5     *RHKNER)**2)
          RHO(KC,JC) = RHOE(KC,JC) / CVTH
        END IF
        RETURN
        END
        SUBROUTINE  DAMPING (J)
        PARAMETER   (MK = 130, MJ = 115)
        COMMON /C/ DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1    CFL, BETA, UINF, RHOINF, CINF, TINF, TW, DT, CX, CY, L, JL, KL,
     2    JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
        COMMON  /DEP/  RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1    MJ)
        COMMON  /DEPP/  RHOP(MK, MJ), RHOUP(MK, MJ), RHOVP(MK, MJ),
     1    RHOEP(MK, MJ)
        COMMON  /DAMP/  ADD1(MK), ADD2(MK), ADD3(MK), ADD5(MK)
        COMMON  /DV/  R(MK, MJ), U(MK, MJ), V(MK, MJ), P(MK, MJ), T(MK,
     1    MJ)
        COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
C
        DIMENSION   ADDG1(MK), ADDG2(MK), ADDG3(MK), ADDG5(MK),
     1    ADDH1(MK), ADDH2(MK), ADDH3(MK), ADDH5(MK), PD(MK, 2)
C
        KS        = KSTART(J)
        KSP2      = KS + 2
        KLM2      = KLM - 1
        GAMMR     = GAMMA * RC
        RDET      = 1.0 / DETA
        RDZT      = 1.0 / DZETA
C
C         SET DAMPING TERMS TO ZERO FOR END POINTS
C
        DO 1 KV   = KS, KL
        ADDG1(KV) = 0.0
        ADDG2(KV) = 0.0
        ADDG3(KV) = 0.0
        ADDG5(KV) = 0.0
C
        ADDH1(KV) = 0.0
        ADDH2(KV) = 0.0
        ADDH3(KV) = 0.0
```

```
      ADDH5(KV) = 0.0
    1 CONTINUE
C
C       GENERATE ADDG
C
      IF ((J .LE. 2) .OR. (J .GE. JLM)) GO TO 7
C
        DO 3 M    = 1, 2
        JV        = J + M + L - 3
        JP        = JV + 1
        JM        = JV - 1
C
          DO 2 KV   = KS, KL
          PD(KV, M) = CY * (ABS(P(KV, JP) - 2.0 * P(KV, JV) + P(KV,
     1      JM)) / (P(KV, JP) + 2.0 * P(KV, JV) + P(KV, JM))) *
     2      (ABS(ETX(KV, JV) * U(KV, JV) + ETY(KV, JV) * V(KV, JV)) +
     3      SQRT(GAMMR * T(KV, JV) * (ETX(KV, JV) ** 2 + ETY(KV, JV) **
     4      2)))
    2     CONTINUE
    3   CONTINUE
C
      IF (L .EQ. 2) GO TO 5
C
C       PREDICTOR
C
        JP        = J + 1
        JM        = J - 1
C
        DO 4 KV   = KS, KL
        ADDG1(KV) = DT * (PD(KV, 2) * (RHO(KV, JP) - RHO(KV, J)) -
     1    PD(KV, 1) * (RHO(KV, J) - RHO(KV, JM))) * RDET
        ADDG2(KV) = DT * (PD(KV, 2) * (RHOU(KV, JP) - RHOU(KV, J)) -
     1    PD(KV, 1) * (RHOU(KV, J) - RHOU(KV, JM))) * RDET
        ADDG3(KV) = DT * (PD(KV, 2) * (RHOV(KV, JP) - RHOV(KV, J)) -
     1    PD(KV, 1) * (RHOV(KV, J) - RHOV(KV, JM))) * RDET
        ADDG5(KV) = DT * (PD(KV, 2) * (RHOE(KV, JP) - RHOE(KV, J)) -
     1    PD(KV, 1) * (RHOE(KV, J) - RHOE(KV, JM))) * RDET
    4   CONTINUE
C
      GO TO 7
C
C       CORRECTOR
C
    5 JP        = J + 1
      JM        = J - 1
C
        DO 6 KV   = KS, KL
        ADDG1(KV) = DT * (PD(KV, 2) * (RHOP(KV, JP) - RHOP(KV, J)) -
     1    PD(KV, 1) * (RHOP(KV, J) - RHOP(KV, JM))) * RDET
        ADDG2(KV) = DT * (PD(KV, 2) * (RHOUP(KV, JP) - RHOUP(KV, J)) -
     1    PD(KV, 1) * (RHOUP(KV, J) - RHOUP(KV, JM))) * RDET
        ADDG3(KV) = DT * (PD(KV, 2) * (RHOVP(KV, JP) - RHOVP(KV, J)) -
     1    PD(KV, 1) * (RHOVP(KV, J) - RHOVP(KV, JM))) * RDET
        ADDG5(KV) = DT * (PD(KV, 2) * (RHOEP(KV, JP) - RHOEP(KV, J)) -
     1    PD(KV, 1) * (RHOEP(KV, J) - RHOEP(KV, JM))) * RDET
    6   CONTINUE
C
C       GENERATE ADDH
C
    7   DO 9 M    = 1, 2
          DO 8 K    = KSP2, KLM2
            KV        = K + M + L - 3
            KP        = KV + 1
            KM        = KV - 1
C
            PD(K, M)  = CX * (ABS(P(KP, J) - 2.0 * P(KV, J) + P(KM, J)) /
     1        (P(KP, J) + 2.0 * P(KV, J) + P(KM, J))) * (ABS(ZTX(KV, J) *
     2        U(KV, J) + ZTY(KV, J) * V(KV, J)) + SQRT(GAMMR * T(KV, J) *
     3        (ZTX(KV, J) ** 2 + ZTY(KV, J) ** 2)))
    8     CONTINUE
    9   CONTINUE
C
      IF (L .EQ. 2) GO TO 11
C
C       PREDICTOR
C
        DO 10 KV  = KSP2, KLM2
        KP        = KV + 1
        KM        = KV - 1
C
        ADDH1(KV) = DT * (PD(KV, 2) * (RHO(KP, J) - RHO(KV, J)) -
     1    PD(KV, 1) * (RHO(KV, J) - RHO(KM, J))) * RDZT
        ADDH2(KV) = DT * (PD(KV, 2) * (RHOU(KP, J) - RHOU(KV, J)) -
     1    PD(KV, 1) * (RHOU(KV, J) - RHOU(KM, J))) * RDZT
        ADDH3(KV) = DT * (PD(KV, 2) * (RHOV(KP, J) - RHOV(KV, J)) -
     1    PD(KV, 1) * (RHOV(KV, J) - RHOV(KM, J))) * RDZT
        ADDH5(KV) = DT * (PD(KV, 2) * (RHOE(KP, J) - RHOE(KV, J)) -
     1    PD(KV, 1) * (RHOE(KV, J) - RHOE(KM, J))) * RDZT
   10   CONTINUE
C
      GO TO 13
C
C       CORRECTOR
C
   11   DO 12 KV  = KSP2, KLM2
        KP        = KV + 1
        KM        = KV - 1
C
        ADDH1(KV) = DT * (PD(KV, 2) * (RHOP(KP, J) - RHOP(KV, J)) -
     1    PD(KV, 1) * (RHOP(KV, J) - RHOP(KM, J))) * RDZT
        ADDH2(KV) = DT * (PD(KV, 2) * (RHOUP(KP, J) - RHOUP(KV, J)) -
     1    PD(KV, 1) * (RHOUP(KV, J) - RHOUP(KM, J))) * RDZT
        ADDH3(KV) = DT * (PD(KV, 2) * (RHOVP(KP, J) - RHOVP(KV, J)) -
     1    PD(KV, 1) * (RHOVP(KV, J) - RHOVP(KM, J))) * RDZT
        ADDH5(KV) = DT * (PD(KV, 2) * (RHOEP(KP, J) - RHOEP(KV, J)) -
     1    PD(KV, 1) * (RHOEP(KV, J) - RHOEP(KM, J))) * RDZT
   12   CONTINUE
C
C       SUM THE ARTIFICIAL VISCOSITY-LIKE TERMS
C
   13   DO 14 KV  = KS, KL
        ADD1(KV)  = ISMTHX * ADDG1(KV) + ISMTHY * ADDH1(KV)
        ADD2(KV)  = ISMTHX * ADDG2(KV) + ISMTHY * ADDH2(KV)
        ADD3(KV)  = ISMTHX * ADDG3(KV) + ISMTHY * ADDH3(KV)
        ADD5(KV)  = ISMTHX * ADDG5(KV) + ISMTHY * ADDH5(KV)
   14   CONTINUE
      IF ((J.EQ.JNERIN).OR.(J.EQ.JNEROUT)) THEN
        DO 15 KV = 1, KNOZ
        ADD1(KV) = 0.0
        ADD2(KV) = 0.0
        ADD3(KV) = 0.0
        ADD5(KV) = 0.0
```

```
   15   CONTINUE
        ENDIF
        RETURN
        END
        SUBROUTINE  EDDY
        PARAMETER   (MK = 130, MJ = 115)
        COMMON /  DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
       1  CFL, BETA, UINF, RHOINF, CINF, TINF, TM, DT, CX, CY, L, JL, KL,
       2  JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
        COMMON  /DEP   RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
       1  MJ)
        COMMON  /DOF   X(MK, MJ), Y(MK, MJ)
        COMMON  /DV    R(MK, MJ), U(MK, MJ), V(MK, MJ), P(MK, MJ), T(MK,
       1  MJ)
        COMMON  /NELSON   RDET, RDZT, QDET, RMU, SL(MK, MJ)
        COMMON  /NOZ   KNOZ, JNOZIN, JNOZOUT, JNWIN, JNWOUT, JNERIN,
       1  JNEROUT, PINOZ, TINOZ, ALPHA, EFFNOZ
        COMMON  /TV    ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
        COMMON  /TVIS  EP(MK, MJ), CPQPR, CPQPRT
C
        DIMENSION   EPMAX(MK), F(MK, MJ), JMAX(MK), OMEGA(MK, MJ),
       1  UET(MK, MJ), UZT(MK, MJ), VET(MK, MJ), VZT(MK, MJ)
C
C       BALDWIN-LOMAX MODEL OF EDDY VISCOSITY WITH INTERMITTENCY
C       CORRECTION.
C       THIS VERSION ASSUMES THE SOLID SURFACE TO BE AT THE TOP, J = JL.
C
        DO 1 J     = 1, JL
          DO 1 K     = 1, KL
          RR         = 1.0 / RHO(K, J)
          U(K, J)    = RHOU(K, J) * RR
          V(K, J)    = RHOV(K, J) * RR
    1     CONTINUE
C
C       GENERATE THE DERIVATIVES OF VELOCITY
C
        DO 2 J     = 2, JLM
          DO 2 K     = 2, KLM
          UET(K, J) = (U(K, J + 1) - U(K, J - 1)) * RDET
          VET(K, J) = (V(K, J + 1) - V(K, J - 1)) * RDET
          UZT(K, J) = (U(K + 1, J) - U(K - 1, J)) * RDZT
          VZT(K, J) = (V(K + 1, J) - V(K - 1, J)) * RDZT
    2     CONTINUE
C
C       GENERATE THE VORTICITY DISTRIBUTION
C
        DO 3 J     = 2, JLM
          DO 3 K     = 2, KLM
          OMEGA(K, J) = SQRT((ETX(K, J) * VET(K, J) + ZTX(K, J) *
       1    VZT(K, J) - ETY(K, J) * UET(K, J) - ZTY(K, J) * UZT(K, J))
       2    ** 2)
    3     CONTINUE
C
C       GENERATE THE SURFACE VORTICITY
C
        DO 4 K     = 2, KLM
        OMEGA(K, JL) = SQRT((ETX(K, JL) * (V(K, JLM) - V(K, JL)) -
       1  ETY(K, JL) * (U(K, JLM) - U(K, JL))) ** 2) * QDET
    4   CONTINUE
C
C       GENERATE THE INNER VISCOSITY COEFFICIENT
C
        KS = KNOZ + 1
        DO 5 J     = 2, JLM
          DO 5 K     = KS, KLM
          YPB        = MIN(SQRT(RHO(K, JL) * OMEGA(K, JL) / RMU) *
       1    SL(K, J) / 26.0, 50.0)
          EP(K, J)   = RHO(K, J) * (0.40 * SL(K, J) * (1.0 - EXP( -
       1    YPB)))** 2 * OMEGA(K, J)
          F(K, J)    = SL(K, J) * (1.0 - EXP( - YPB)) * OMEGA(K, J)
    5     CONTINUE
C
C       SEARCH THE MAX. EDDY VISCOSITY COEFF. FOR A J.
C
        DO 7 K     = KS, KLM
        DU2MAX     = 0.0
        FMAX       = 0.0
        JMAX(K)    = 0
          DO 6 J     = JLM, 2, -1
          DU2MAX     = MAX(U(K, J) ** 2 + V(K, J) ** 2, DU2MAX)
          IF (F(K,J) .GT. FMAX) THEN
            FMAX       = F(K, J)
            JMAX(K)    = J
          END IF
    6     CONTINUE
C
        FWAKE      = SL(K, JMAX(K)) * F(K, JMAX(K))
        F1         = 0.25 * SL(K, JMAX(K)) * DU2MAX / F(K, JMAX(K))
        FWAKE      = MIN(FWAKE, F1)
        EPMAX(K)   = 0.0336 * RHO(K, JMAX(K)) * FWAKE
    7   CONTINUE
C
C       GENERATE SURFACE VORTICITY ON NOZZLE WALLS
C
        DO 8 K     = 2, KNOZ
        OMEGA(K, JNWIN) = SQRT((ETX(K, JNWIN) * (V(K, JNERIN) -
       1  V(K, JNWIN)) - ETY(K, JNWIN) * (U(K, JNERIN) -
       2  U(K, JNWIN)))** 2) * QDET
    8   CONTINUE
        DO 9 K     = 2, KNOZ
        OMEGA(K, JNWOUT) = SQRT((ETX(K, JNWOUT) * (V(K, JNEROUT)
       1  - V(K, JNWOUT)) - ETY(K, JNWOUT) * (U(K, JNEROUT) -
       2  U(K, JNWOUT)))** 2) * QDET
    9   CONTINUE
C
C       FIND INNER VISCOSITY COEFFICIENT FOR K = 2, KNOZ
C
        JCL = JNWOUT + (JL - JNWOUT) / 2
        DO 12 K    = 2, KNOZ
          DO 10 J    = 2, JNERIN
          YPB = MIN(SQRT(RHO(K, JNWIN) * OMEGA(K, JNWIN) / RMU) *
       1    SL(K, J) / 26.0, 50.0)
          EP(K, J)   = RHO(K, J) * (0.40 * SL(K, J) * (1.0 - EXP( -
       1    YPB)))** 2 * OMEGA(K, J)
          F(K, J)    = SL(K, J) * (1.0 - EXP( - YPB)) * OMEGA(K, J)
   10     CONTINUE
          DO 11 J    = JNEROUT, JCL
          YPB = MIN(SQRT(RHO(K, JNWOUT) * OMEGA(K, JNWOUT) / RMU)
       1    * SL(K, J) / 26.0, 50.0)
          EP(K, J)   = RHO(K, J) * (0.40 * SL(K, J) * (1.0 - EXP( -
       1    YPB)))** 2 * OMEGA(K, J)
          F(K, J)    = SL(K, J) * (1.0 - EXP( - YPB)) * OMEGA(K, J)
```

```
   11     CONTINUE
          DO 12 J    = JCL, JLM
          YPB = MIN(SQRT(RHO(K, JL) * OMEGA(K, JL) / RMU)
     1      * SL(K, J) / 26.0, 50.0)
          EP(K, J)   = RHO(K, J) * (0.40 * SL(K, J) * (1.0 - EXP( -
     1      YPB)) ** 2 * OMEGA(K, J)
          F(K, J)    = SL(K, J) * (1.0 - EXP( - YPB)) * OMEGA(K, J)
   12     CONTINUE
C
C         FIND MAX EDDY VISCOSITY COEFF FOR K = 2, KNOZ, J .GT. JNWOUT
C
          DO 14 K    = 2, KNOZ
          DU2MAX     = 0.0
          FMAX       = 0.0
          JMAX(K)    = 0
            DO 13 J    = JLM, JNEROUT, -1
            DU2MAX     = MAX(U(K, J) ** 2 + V(K, J) ** 2, DU2MAX)
            IF (F(K,J) .GT. FMAX) THEN
              FMAX       = F(K, J)
              JMAX(K)    = J
            END IF
   13       CONTINUE
C
          FWAKE      = SL(K, JMAX(K)) * F(K, JMAX(K))
          F1         = 0.25 * SL(K, JMAX(K)) * DU2MAX / F(K, JMAX(K))
          FWAKE      = MIN(FWAKE, F1)
          EPMAX(K)   = 0.0336 * RHO(K, JMAX(K)) * FWAKE
   14     CONTINUE
C
C         SET THE MAX EDDY VISCOSITY FOR THE OUTER REGION
C
          DO 16 K    = 2, KNOZ
          ISW        = 0
            DO 15 J    = JLM, JNEROUT, -1
            IF (EPMAX(K) .LE. EP(K,J)) ISW = 1
            IF (ISW .GT. 0) EP(K, J) = EPMAX(K) * RHO(K, J) /
     1        RHO(K, JMAX(K))
            EP(K, J)   = EP(K, J) / (1.0 + 5.5 * (0.3 * SL(K, J) / SL(K,
     1        JMAX(K))) ** 6)
   15       CONTINUE
   16     CONTINUE
          DO 18 K    = KS, KLM
          ISW        = 0
            DO 17 J    = JLM, 2, -1
            IF (EPMAX(K) .LE. EP(K,J)) ISW = 1
            IF (ISW .GT. 0) EP(K, J) = EPMAX(K) * RHO(K, J) /
     1        RHO(K, JMAX(K))
            EP(K, J)   = EP(K, J) / (1.0 + 5.5 * (0.3 * SL(K, J) / SL(K,
     1        JMAX(K))) ** 6)
   17       CONTINUE
   18     CONTINUE
C
C         FIND MAX EDDY VISCOSITY COEFF FOR K = 2, KNOZ, J .LT. JNWIN
C
          DO 20 K = 2, KNOZ
          DU2MAX     = 0.0
          FMAX       = 0.0
          JMAX(K)    = 0
            DO 19 J    = JNERIN, 2, -1
            DU2MAX     = MAX(U(K, J) ** 2 + V(K, J) ** 2, DU2MAX)
            IF (F(K,J) .GT. FMAX) THEN
              FMAX       = F(K, J)
              JMAX(K)    = J
            END IF
   19       CONTINUE
C
          FWAKE      = SL(K, JMAX(K)) * F(K, JMAX(K))
          F1         = 0.25 * SL(K, JMAX(K)) * DU2MAX / F(K, JMAX(K))
          FWAKE      = MIN(FWAKE, F1)
          EPMAX(K)   = 0.0336 * RHO(K, JMAX(K)) * FWAKE
   20     CONTINUE
C
C         SET THE MAX EDDY VISCOSITY FOR THE OUTER REGION
C
          DO 22 K    = 2, KNOZ
          ISW        = 0
            DO 21 J    = JNERIN, 2, -1
            IF (EPMAX(K) .LE. EP(K,J)) ISW = 1
            IF (ISW .GT. 0) EP(K, J) = EPMAX(K) * RHO(K, J) /
     1        RHO(K, JMAX(K))
            EP(K, J)   = EP(K, J) / (1.0 + 5.5 * (0.3 * SL(K, J) / SL(K,
     1        JMAX(K))) ** 6)
   21       CONTINUE
   22     CONTINUE
C
C         COMPLETE THE EDDY VISCOSITY MATRIX
C
          EP(1, 2) = EP(2, 2)
          EP(KL, 2) = EP(KLM, 2)
C
            DO 23 K    = 1, KL
            EP(K, JL) = 0.0
   23       EP(K, 1)   = EP(K, 2)
C
            DO 24 J    = 1, JL
            EP(KL, J) = EP(KLM, J)
   24       EP(1, J)   = EP(2, J)
C
            DO 25 K = 1, KNOZ
              DO 25 J = JNWIN, JNWOUT
              EP(K, J) = 0.0
   25       CONTINUE
          RETURN
          END
          SUBROUTINE  LETA (J)
          PARAMETER   (MK = 130, MJ = 115)
          COMMON // DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1      CFL, BETA, UINF, RHOINF, CINF, TINF, TW, DT, CX, CY, L, JL, KL,
     2      JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
          COMMON  /DOF/  XI(MK, MJ), Y(MK, MJ), RY(MK, MJ)
          COMMON  /FLUET/  F21(MK, 2), F22(MK, 2), F23(MK, 2), F25(MK, 2),
     1      G21(MK, 2), G22(MK, 2), G23(MK, 2), G25(MK, 2), H(MK)
          COMMON  /DV/  R(MK, MJ), U(MK, MJ), V(MK, MJ), P(MK, MJ), T(MK,
     1      MJ)
          COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
          COMMON  /TVIS/  EP(MK, MJ), CPQPR, CPQPRT
C
          DIMENSION   DUDX(MK), DVDR(MK), RK(MK), RLMBD(MK), RMU(MK),
     1      SMU(MK), TAURR(MK), TAUXR(MK), TAUXX(MK), TET(MK), TX(MK),
     2      TY(MK), TZT(MK), UET(MK), UZT(MK), VET(MK), VZT(MK)
C
          RDET       = 1.0 / DETA
```

```
      RDZT      = 1.0 / (2.0 * DZETA)
      KSP       = 2
C
        DO 7 M    = 1, 2
C
CC       FORWARD & BACKWARD DIFFERENCING FOR PREDICTOR & CORRECTOR
C
        JV        = L + M + J - 3
        JM        = J + M - 2
        JP        = JM + 1
C
          DO 1 KV   = KSP, KLM
          KP        = KV + 1
          KM        = KV - 1
C
CC       GENERATE THE SHEAR STRESS & HEAT FLUX TERMS
C
          UET(KV)   = (U(KV, JP) - U(KV, JM)) * RDET
          VET(KV)   = (V(KV, JP) - V(KV, JM)) * RDET
          TET(KV)   = (T(KV, JP) - T(KV, JM)) * RDET
C
          UZT(KV)   = (U(KP, JV) - U(KM, JV)) * RDZT
          VZT(KV)   = (V(KP, JV) - V(KM, JV)) * RDZT
          TZT(KV)   = (T(KP, JV) - T(KM, JV)) * RDZT
C
CC       GENERATE THE VISCOSITY COEFFS & HEAT CONDUCTIVITY
C
          RMU(KV)   = 2.27E - 08 * SQRT(T(KV, JV) ** 3) / (T(KV, JV) +
     1      198.6)
          RK(KV)    = CPQPR * RMU(KV) + CPQPRT * EP(KV, JV)
          RMU(KV)   = RMU(KV) + EP(KV, JV)
          RLMBD(KV) = - (2.0 / 3.0) * RMU(KV)
          SMU(KV)   = 2.0 * RMU(KV) + RLMBD(KV)
     1    CONTINUE
C
          DO 2 KV   = KSP, KLM
          DUDX(KV)  = ETX(KV, JV) * UET(KV) + ZTX(KV, JV) * UZT(KV)
          DVDR(KV)  = ETY(KV, JV) * VET(KV) + ZTY(KV, JV) * VZT(KV)
C
          TAUXX(KV) = SMU(KV) * DUDX(KV) + RLMBD(KV) * (V(KV, JV) *
     1      RY(KV, JV) + DVDR(KV)) - P(KV, JV)
          TAURR(KV) = SMU(KV) * DVDR(KV) + RLMBD(KV) * (V(KV, JV) *
     1      RY(KV, JV) + DUDX(KV)) - P(KV, JV)
          TAUXR(KV) = RMU(KV) * (ETY(KV, JV) * UET(KV) + ZTY(KV, JV) *
     1      UZT(KV) + ETX(KV, JV) * VET(KV) + ZTX(KV, JV) * VZT(KV))
     2    CONTINUE
C
        IF (JV .NE. J) GO TO 4
C
        JP        = J + 1
        JM        = J - 1
          DO 3 KV   = KSP, KLM
          KVP       = KV + 1
          KVM       = KV - 1
C
          DUDX(KV)  = ETX(KV, J) * (U(KV, JP) - U(KV, JM)) * (0.5 *
     1      RDET) + ZTX(KV, J) * (U(KVP, J) - U(KVM, J)) * RDZT
          DVDR(KV)  = ETY(KV, J) * (V(KV, JP) - V(KV, JM)) * (0.5 *
     1      RDET) + ZTY(KV, J) * (V(KVP, J) - V(KVM, J)) * RDZT
C
          H(KV)     = (SMU(KV) * V(KV, J) * RY(KV, J) + RLMBD(KV) *
     1      (DUDX(KV) + DVDR(KV)) - P(KV, J)) * RY(KV, J)
     3    CONTINUE
C
CC       GENERATE THE HEAT FLUX TERMS
C
     4    DO 5 KV   = KSP, KLM
          TX(KV)    = ETX(KV, JV) * TET(KV) + ZTX(KV, JV) * TZT(KV)
          TY(KV)    = ETY(KV, JV) * TET(KV) + ZTY(KV, JV) * TZT(KV)
     5    CONTINUE
C
CC       GENERATE THE FLUX TERMS
C
          DO 6 KV   = KSP, KLM
          F21(KV, M) = R(KV, JV) * U(KV, JV)
          F22(KV, M) = F21(KV, M) * U(KV, JV) - TAUXX(KV)
          F23(KV, M) = F21(KV, M) * V(KV, JV) - TAUXR(KV)
          F25(KV, M) = F21(KV, M) * (CV * T(KV, JV) + 0.5 * (U(KV, JV)
     1      ** 2 + V(KV, JV) ** 2)) - RK(KV) * TX(KV) - (U(KV, JV) *
     2      TAUXX(KV) + V(KV, JV) * TAUXR(KV))
C
          G21(KV, M) = R(KV, JV) * V(KV, JV) * Y(KV, JV)
          G22(KV, M) = G21(KV, M) * U(KV, JV) - TAUXR(KV) * Y(KV, JV)
          G23(KV, M) = G21(KV, M) * V(KV, JV) - TAURR(KV) * Y(KV, JV)
          G25(KV, M) = G21(KV, M) * (CV * T(KV, JV) + 0.5 * (U(KV, JV)
     1      ** 2 + V(KV, JV) ** 2)) - (RK(KV) * TY(KV) + (U(KV, JV) *
     2      TAUXR(KV) + V(KV, JV) * TAURR(KV))) * Y(KV, JV)
     6    CONTINUE
     7  CONTINUE
      RETURN
      END
      SUBROUTINE  LZETA (J)
      PARAMETER   (MK = 130, MJ = 115)
      COMMON /  DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1  CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
     2  JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
      COMMON  /DOF/  X(MK, MJ), Y(MK, MJ), RY(MK, MJ)
      COMMON  /FLUZT/  F31(MK), F32(MK), F33(MK), F35(MK), G31(MK),
     1  G32(MK), G33(MK), G35(MK)
      COMMON  /DV/  R(MK, MJ), U(MK, MJ), V(MK, MJ), P(MK, MJ), T(MK,
     1  MJ)
      COMMON  /MONC/  G31N(2), G32N(2), G33N(2), G35N(2)
      COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
      COMMON  /TVIS/  EP(MK, MJ), CPQPR, CPQPRT
C
      DIMENSION   DUDX(MK), DVDR(MK), RK(MK), RLMBD(MK), RMU(MK),
     1  SMU(MK), TAURR(MK), TAUXR(MK), TAUXX(MK), TET(MK), TX(MK),
     2  TY(MK), TZT(MK), UET(MK), UZT(MK), VET(MK), VZT(MK)
C
      RDET      = 1.0 / (2.0 * DETA)
      RDZT      = 1.0 / DZETA
      JP        = J + 1
      JM        = J - 1
      KS        = 1
      KSP       = KS + 1
C
        DO 1 KK   = KS, KLM
C
CC       FORWARD & BACKWARD DIFFERENCING FOR PREDICTOR & CORRECTOR
C
        KV        = KK + L - 1
        KM        = KK
```

```
      KP        = KM + 1
C
C
C       GENERATE THE SHEAR STRESS & HEAT FLUX TERMS
C
        UET(KK)   = (U(KV, JP) - U(KV, JM)) * RDET
        VET(KK)   = (V(KV, JP) - V(KV, JM)) * RDET
        TET(KK)   = (T(KV, JP) - T(KV, JM)) * RDET
C
        UZT(KK)   = (U(KP, J) - U(KM, J)) * RDZT
        VZT(KK)   = (V(KP, J) - V(KM, J)) * RDZT
        TZT(KK)   = (T(KP, J) - T(KM, J)) * RDZT
C
C
C       GENERATE THE VISCOSITY COEFFS & HEAT CONDUCTIVITY
C
        RMU(KK)   = 2.27E - 08 * SQRT(T(KV, J) ** 3) / (T(KV, J) +
     1    198.6)
        RK(KK)    = CPQPR * RMU(KK) + CPQPRT * EP(KV, J)
        RMU(KK)   = RMU(KK) + EP(KV, J)
        RLMBD(KK) = - (2.0 / 3.0) * RMU(KK)
        SMU(KK)   = 2.0 * RMU(KK) + RLMBD(KK)
    1   CONTINUE
C
      IF (L .NE. 1) GO TO 2
C
C
C       CALCULATE TX, TY FOR L = 1, KK = 1, KV = 1
C       USING L'HOSPITAL RULE, V / R = DVDR WHEN R = 0
C
        DUDX(1)   = ETX(1, J) * UET(1) + ZTX(1, J) * UZT(1)
        DVDR(1)   = ETY(1, J) * VET(1) + ZTY(1, J) * VZT(1)
        TAUXX(1)  = SMU(1) * DUDX(1) + RLMBD(1) * (DVDR(1) + DVDR(1)) -
     1    P(1, J)
        TAURR(1)  = SMU(1) * DVDR(1) + RLMBD(1) * (DVDR(1) + DUDX(1)) -
     1    P(1, J)
        TAUXR(1)  = RMU(1) * (ETY(1, J) * UET(1) + ZTY(1, J) * UZT(1) +
     1    ETX(1, J) * VET(1) + ZTX(1, J) * VZT(1))
C
C
C       GENERATE HEAT FLUX TERMS
C
        TX(1)     = ETX(1, J) * TET(1) + ZTX(1, J) * TZT(1)
        TY(1)     = ETY(1, J) * TET(1) + ZTY(1, J) * TZT(1)
        GO TO 3
C
C
C       CALCULATE TX, TY, FOR L = 2, KK = 1, KV = 2
C
    2   DUDX(1)   = ETX(2, J) * UET(1) + ZTX(2, J) * UZT(1)
        DVDR(1)   = ETY(2, J) * VET(1) + ZTY(2, J) * VZT(1)
        TAUXX(1)  = SMU(1) * DUDX(1) + RLMBD(1) * (V(2, J) / Y(2, J) +
     1    DVDR(1)) - P(2, J)
        TAURR(1)  = SMU(1) * DVDR(1) + RLMBD(1) * (V(2, J) / Y(2, J) +
     1    DUDX(1)) - P(2, J)
        TAUXR(1)  = RMU(1) * (ETY(2, J) * UET(1) + ZTY(2, J) * UZT(1) +
     1    ETX(2, J) * VET(1) + ZTX(2, J) * VZT(1))
C
C
C       GENERATE HEAT FLUX TERMS
C
        TX(1)     = ETX(2, J) * TET(1) + ZTX(2, J) * TZT(1)
        TY(1)     = ETY(2, J) * TET(1) + ZTY(2, J) * TZT(1)
C
    3   DO 4 KK   = KSP, KLM
        KV        = KK + L - 1
C
        DUDX(KK)  = ETX(KV, J) * UET(KK) + ZTX(KV, J) * UZT(KK)
        DVDR(KK)  = ETY(KV, J) * VET(KK) + ZTY(KV, J) * VZT(KK)
C
        TAUXX(KK) = SMU(KK) * DUDX(KK) + RLMBD(KK) * (V(KV, J) * RY(KV,
     1    J) + DVDR(KK)) - P(KV, J)
        TAURR(KK) = SMU(KK) * DVDR(KK) + RLMBD(KK) * (V(KV, J) * RY(KV,
     1    J) + DUDX(KK)) - P(KV, J)
        TAUXR(KK) = RMU(KK) * (ETY(KV, J) * UET(KK) + ZTY(KV, J) *
     1    UZT(KK) + ETX(KV, J) * VET(KK) + ZTX(KV, J) * VZT(KK))
C
C
C       GENERATE THE HEAT FLUX TERMS
C
        TX(KK)    = ETX(KV, J) * TET(KK) + ZTX(KV, J) * TZT(KK)
        TY(KK)    = ETY(KV, J) * TET(KK) + ZTY(KV, J) * TZT(KK)
    4   CONTINUE
C
C
C       GENERATE THE FLUX TERMS
C
        DO 5 KK   = KS, KLM
        KV        = KK + L - 1
C
        F31(KK)   = R(KV, J) * U(KV, J)
        F32(KK)   = F31(KK) * U(KV, J) - TAUXX(KK)
        F33(KK)   = F31(KK) * V(KV, J) - TAUXR(KK)
        F35(KK)   = F31(KK) * (CV * T(KV, J) + 0.5 * (U(KV, J) ** 2 +
     1    V(KV, J) ** 2)) - RK(KK) * TX(KK) - (U(KV, J) * TAUXX(KK) +
     2    V(KV, J) * TAUXR(KK))
C
        G31(KK)   = R(KV, J) * V(KV, J) * Y(KV, J)
        G32(KK)   = G31(KK) * U(KV, J) - TAUXR(KK) * Y(KV, J)
        G33(KK)   = G31(KK) * V(KV, J) - TAURR(KK) * Y(KV, J)
        G35(KK)   = G31(KK) * (CV * T(KV, J) + 0.5 * (U(KV, J) ** 2 +
     1    V(KV, J) ** 2)) - (RK(KK) * TY(KK) + (U(KV, J) * TAUXR(KK) +
     2    V(KV, J) * TAURR(KK))) * Y(KV, J)
    5   CONTINUE
C
      IF (L .EQ. 2) GO TO 7
C
C       NON-CONSERVATIVE FORM IS USED ONLY FOR L = 1.
C
        DO 6 KK   = KS, KSP
        KV        = KK
        G31N(KK)  = R(KV, J) * V(KV, J)
        G32N(KK)  = G31N(KK) * U(KV, J) - TAUXR(KK)
        G33N(KK)  = G31N(KK) * V(KV, J) - TAURR(KK)
        G35N(KK)  = G31N(KK) * (CV * T(KV, J) + 0.5 * (U(KV, J) ** 2 +
     1    V(KV, J) ** 2)) - (RK(KK) * TY(KK) + (U(KV, J) * TAUXR(KK) +
     2    V(KV, J) * TAURR(KK)))
    6   CONTINUE
    7 RETURN
      END
      SUBROUTINE  L2NORM
      PARAMETER   (MK = 130, MJ = 115)
      COMMON /I/ DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1   CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
     2   JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
      COMMON  /CGAS/  CVTH, CVTAH, GAMMP, GAMMR, GAMMT, RUINF, REINF
      COMMON  /DEP/  RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1   MJ)
      COMMON  /OLD/  ROLD(MK, MJ), RUOLD(MK, MJ), RVOLD(MK, MJ),
     1   REOLD(MK, MJ), XNORM
```

```
      COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
C
        DO 1 J    = 1, JL
          DO 1 K    = 1, KL
          ROLD(K, J) = ABS(RHO(K, J) - ROLD(K, J)) / RHOINF
          RUOLD(K, J) = ABS(RHOU(K, J) - RUOLD(K, J)) / RUINF
          RVOLD(K, J) = ABS(RHOV(K, J) - RVOLD(K, J)) / RUINF
          REOLD(K, J) = ABS(RHOE(K, J) - REOLD(K, J)) / REINF
    1     CONTINUE
        XNORM     = 0.0
          DO 2 J    = 1, JL
            DO 2 K    = 1, KL
            XNORM     = XNORM + ROLD(K, J) ** 2 + RUOLD(K, J) ** 2 +
     1        RVOLD(K, J) ** 2 + REOLD(K, J) ** 2
     2      CONTINUE
        XNORM     = SQRT(XNORM)
C
C         RESET FOR NEXT TIME.
C
          DO 3 J    = 1, JL
            DO 3 K    = 1, KL
            ROLD(K, J) = RHO(K, J)
            RUOLD(K, J) = RHOU(K, J)
            RVOLD(K, J) = RHOV(K, J)
            REOLD(K, J) = RHOE(K, J)
    3       CONTINUE
        RETURN
        END
        SUBROUTINE  PAGE (OOPS)
        PARAMETER   (MK = 130, MJ = 115)
        COMMON // DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1    CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
     2    JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
        COMMON  /DV/  R(MK, MJ), U(MK, MJ), V(MK, MJ), P(MK, MJ), T(MK,
     1    MJ)
        COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
        COMMON  /DEP/  RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1    MJ)
        COMMON  /DEPP/  RHOP(MK, MJ), RHOUP(MK, MJ), RHOVP(MK, MJ),
     1    RHOEP(MK, MJ)
        COMMON  /ERR/  KFE, JFE
C
C         PREDICTOR SWEEP
C
        LOGICAL      OOPS
C
        OOPS      = .FALSE.
        L         = 1
C
C         GENERATE THREE PAGES OF TEMPORAL VECTOR ARRAYS
C
          DO 1 JV   = 1, JL
            DO 1 KV   = 1, KL
            R(KV, JV) = RHO(KV, JV)
            U(KV, JV) = RHOU(KV, JV) / R(KV, JV)
            V(KV, JV) = RHOV(KV, JV) / R(KV, JV)
            T(KV,JV) = (RHOE(KV,JV)/R(KV,JV) - 0.5 * (U(KV,JV)**2
     1        + V(KV,JV)**2))/CV
            IF (T(KV,JV) .LT. 25.0) THEN
              OOPS      = .TRUE.
              KFE       = KV
              JFE       = JV
              RETURN
            END IF
            P(KV, JV) = R(KV, JV) * RC * T(KV, JV)
    1       CONTINUE
C
          DO 2 J    = 2, JLM
          CALL LETA (J)
          CALL LZETA (J)
          CALL SUM (J)
    2     CONTINUE
C
        CALL BC
C
C         CORRECTOR SWEEP
C
        L         = 2
C
C         GENERATE THREE PAGES OF TEMPORAL VECTOR ARRAYS
C
          DO 3 JV   = 1, JL
            DO 3 KV   = 1, KL
            R(KV, JV) = RHOP(KV, JV)
            U(KV, JV) = RHOUP(KV, JV) / R(KV, JV)
            V(KV, JV) = RHOVP(KV, JV) / R(KV, JV)
            T(KV,JV) = (RHOEP(KV,JV)/R(KV,JV) - 0.5*(U(KV,JV)**
     1        2 + V(KV,JV)**2))/CV
            IF (T(KV,JV) .LT. 25.0) THEN
              OOPS      = .TRUE.
              KFE       = KV
              JFE       = JV
              RETURN
            END IF
            P(KV, JV) = R(KV, JV) * RC * T(KV, JV)
    3       CONTINUE
C
          DO 4 J    = 2, JLM
          CALL LETA (J)
          CALL LZETA (J)
          CALL SUM (J)
    4     CONTINUE
        CALL BC
        RETURN
        END
        SUBROUTINE  PLOTCN
        PARAMETER   (MK = 130, MJ = 115)
        COMMON // DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1    CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
     2    JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
        COMMON  /DEP/  RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1    MJ)
        COMMON  /DOF/  X(MK, MJ), Y(MK, MJ), RY(MK, MJ)
        COMMON  /DV/  R(MK, MJ), U(MK, MJ), V(MK, MJ), P(MK, MJ), T(MK,
     1    MJ)
        COMMON  /NOZ/  KNOZ, JNOZIN, JNOZOUT, JNWIN, JNWOUT, JNERIN,
     1    JNEROUT, PTNOZ, TTNOZ, ALPHA, EFFNCZ
        COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
        COMMON  /PLT/  RL, RH, RCV, PINF, KSTEP
C
C         WRITE OUT THE COMPUTED FLOW DATA
C
```

```
      DIMENSION   ARCL(MK), CF1(MK), CF2(MK), PBYPI(MK), RMUH(MK),
     1  STANT1(MK), STANT2(MK), XBBP(MK), YBBP(MK), XBBPI(KNOZ),
     2  YBBPI(KNOZ), XBBPO(KNOZ), YBBPO(KNOZ), CF2I(KNOZ),
     3  CF2O(KNOZ), RMUMI(KNOZ), RMUMO(KNOZ)
C
      JLL       = JLM - 1
      GAR       = GAMMA * RC
      RUINF     = 1.0 / UINF
C
      DO 2 K    = 1, KL
      IF (K .EQ. 1 .OR. K .EQ. KL .OR. MOD(K,KSTEP) .EQ. 0) THEN
        WRITE (6,  100) K
          DO 1 J    = 1, JL
          RH        = 1.0 / RHO(K, J)
          UU        = RHOU(K, J) * RH
          VV        = RHOV(K, J) * RH
          RR        = RHO(K, J) / RHOINF
          UR        = UU * RUINF
          VR        = VV * RUINF
          TT        = (RHOE(K, J) * RH - 0.5 * (UU ** 2 + VV ** 2)) *
     1              RCV
          PP        = RHO(K, J) * RC * TT
          TR        = TT / TINF
          RP        = PP / PINF
          QCS       = 1.0 / SQRT(GAR * TT)
          XM        = SQRT(UU ** 2 + VV ** 2) * QCS
          AXM       = UU * QCS
C
          WRITE (6,  101) J  X (K, J), Y (K, J), RR, UR, AXM, VR, XM,
     1          TR, RP
    1     CONTINUE
      END IF
      PBYPI(K)  = RHO(K, JL) * RC * TW / PINF
    2 CONTINUE
C
      ARCL(1)   = 0.0
        DO 3 K    = 1, KL
        XBBP(K)   = X(K, JL)
        YBBP(K)   = Y(K, JL)
    3   CONTINUE
C
        DO 4 K    = 2, KL
        DELS      = SQRT((XBBP(K) - XBBP(K - 1)) ** 2 + (YBBP(K) -
     1    YBBP(K - 1)) ** 2)
        ARCL(K)   = ARCL(K - 1) + DELS
    4   CONTINUE
C
C       CALCULATE STANTON NUMBER
C
      CP        = GAMMA * CV
      STETHY    = CP * TINF + (0.5 * UINF * UINF)
      ENTHWL    = CP * TW
      DENS      = PR * RHOINF * UINF * (STETHY - ENTHWL)
      DENF      = 0.5 * (RHOINF * UINF * UINF)
      WRITE (6,  102)
C
C       FIRST ORDER APPROXIMATION
C
C
        DO 5 K    = 1, KL
        DERE1     = ((RHOE(K, JL) / RHO(K, JL)) - (RHOE(K, JLM) /
     1    RHO(K, JLM))) / DETA
```

```
      RMUH(K)   = 2.27E - 08 * SQRT(T(K, JLM) ** 3) / (T K, JLM) +
     1    198.6)
      ANUM1     = GAMMA * RMUH(K) * (SQRT(ETX(K, JLM) ** 2 + ETY(K,
     1    JLM) ** 2)) * DERE1
      STANT1(K) = - ANUM1 / DENS
C
C       SECOND ORDER APPROXIMATION
C
      DERE2     =  - ((4.0 * RHOE(K, JLM) / RHO(K, JLM)) - (RHOE(K,
     1    JLL) / RHO(K, JLL)) - (3.0 * RHOE(K, JL) / RHO(K, JL))) /
     2    (2.0 * DETA)
      ANUM2     = GAMMA * RMUH(K) * (SQRT(ETX(K, JL) ** 2 + ETY(K,
     1    JL) ** 2)) * DERE2
      STANT2(K) = - ANUM2 / DENS
C
C       CALCULATE SKIN FRICTION COEFFICIENT
C
C       FIRST ORDER APPROXIMATION
C
      DERU1     = ((RHOU(K, JL) / RHO(K, JL)) - (RHOU(K, JLM) /
     1    RHO(K, JLM))) / DETA
      DERV1     = ((RHOV(K, JL) / RHO(K, JL)) - (RHOV(K, JLM) /
     1    RHO(K, JLM))) / DETA
      ANUM1     = RMUH(K) * (ETY(K, JLM) * DERU1 - ETX(K, JLM) *
     1    DERV1)
      CF1(K)    = - ANUM1 / DENF
C
C       SECOND ORDER APPROXIMATION
C
      DERU2     =  - ((4.0 * RHOU(K, JLM) / RHO(K, JLM)) - (RHOU(K,
     1    JLL) / RHO(K, JLL)) - (3.0 * RHOU(K, JL) / RHO(K, JL))) /
     2    (2.0 * DETA)
      DERV2     =  - ((4.0 * RHOV(K, JLM) / RHO(K, JLM)) - (RHOV(K,
     1    JLL) / RHO(K, JLL)) - (3.0 * RHOV(K, JL) / RHO(K, JL))) /
     2    (2.0 * DETA)
      ANUM2     = RMUH(K) * (ETY(K, JL) * DERU2 - ETX(K, JL) * DERV2
      CF2(K)    = - ANUM2 / DENF
    5 CONTINUE
C
      DO 6 K    = 1, KL
      WRITE (6,  103) XBBP (K), CF1 (K), CF2 (K), STANT1 (K), STANT2
     1    (K), PBYPI (K), YBBP (K)
    6 CONTINUE
      WRITE (6, 104)
      JIM = JNWIN - 1
      JIM2 = JIM - 1
      JOP = JNWOUT + 1
      JOP2 = JOP + 1
      DO 7 K    = 1, KNOZ
      XBBPI(K)  = X(K, JNWIN)
      YBBPI(K)  = Y(K, JNWIN)
      XBBPO(K)  = X(K, JNWOUT)
      YBBPO(K)  = Y(K, JNWOUT)
C
C       CALCULATE SKIN FRICTION COEFFICIENT
C
C       SECOND ORDER APPROXIMATION
C
      RMUMI(K)  = 2.27E - 08 * SQRT(T(K, JIM) ** 3) /
     1    (T(K, JIM) + 198.6)
      RMUMO(K)  = 2.27E - 08 * SQRT(T(K, JOP) ** 3) /
```

```
     1      (T(K, JOP) + 198.6)
            DERU2I  =  - ( 4.0 * RHOU(K, JIM) / RHO(K, JIM) ) - (RHOU(K,
     1        JIM2) / RHO(K, JIM2)) - (3.0 * RHOU(K, JNHIN) /
     2        RHO(K, JNHIN)) ) / (2.0 * DETA)
            DERU2O  =  - ( 4.0 * RHOU(K, JOP) / RHO(K, JOP) ) - (RHOU(K,
     1        JOP2) / RHO(K, JOP2)) - (3.0 * RHOU(K, JNHOUT) /
     2        RHO(K, JNHOUT)) ) / (2.0 * DETA)
            DERV2I  =  - ( 4.0 * RHOV(K, JIM) / RHO(K, JIM) ) - (RHOV(K,
     1        JIM2) / RHO(K, JIM2)) - (3.0 * RHOV(K, JNHIN) /
     2        RHO(K, JNHIN)) ) / (2.0 * DETA)
            DERV2O  =  - ( 4.0 * RHOV(K, JOP) / RHO(K, JOP) ) - (RHOV(K,
     1        JOP2) / RHO(K, JOP2)) - (3.0 * RHOV(K, JNHOUT) /
     2        RHO(K, JNHOUT)) ) / (2.0 * DETA)
            ANUM2I = RHUHI(K) * (ETY(K, JNHIN) * DERU2I -
     1        ETX(K, JNHIN) * DERV2I)
            ANUM2O = RHUHO(K) * (ETY(K, JNHOUT) * DERU2O -
     1        ETX(K, JNHOUT) * DERV2O)
            CF2I(K)    = - ANUM2I / DENF
            CF2O(K)    = - ANUM2O / DENF
            WRITE (6, 105) XBBPI(K), YBBPI(K), CF2I(K), XBBPO(K),
     1        YBBPO(K), CF2O(K)
     7    CONTINUE
          RETURN
C
   100  FORMAT ( 'O   K =', I4 / 4X, 'J', 8X, 'X', 10X, 'R', 5X,
     1    'RHO/RHOREF', 3X, 'U/UREF', 5X, 'AXMACH', 5X, 'V/UREF', 9X,
     2    'MACH', 5X, 'T/TREF', 5X, 'P/PREF' )
   101  FORMAT ( 2X, I3, 2X, 2G11.5, G13.5, 3G11.5, G13.5, 2G11.5 )
   102  FORMAT ( 'O', 10X, 'X', 12X, 'CFI', 12X, 'CF2', 10X, 'STNO1',
     1    10X, 'STNO2', 10X, 'P/PREF', 12X, 'Y' )
   103  FORMAT ( 7E15.5 )
   104  FORMAT ( 'O', 8X, 'X', 14X, 'Y', 14X, 'CF2', 12X, 'X', 14X,
     1    'Y', 14X, 'CF2' )
   105  FORMAT ( 6E15.5 )
C
          END
          SUBROUTINE  PREMB
          PARAMETER   (MK = 130, MJ = 115)
          COMMON // DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1      CFL, BETA, UINF, RHOINF, CINF, TINF, TW, DT, CX, CY, L, JL, KL,
     2      JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
          COMMON  /CGAS/  CVTW, CVTAW, GAMMP, GAMMR, GAMMT, RUINF, REINF
          COMMON  /DEP/  RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1      MJ)
          COMMON  /DEPP/  RHOP(MK, MJ), RHOUP(MK, MJ), RHOVP(MK, MJ),
     1      RHOEP(MK, MJ)
          COMMON  /DOF/  X(MK, MJ), Y(MK, MJ), RY(MK, MJ)
          COMMON  /DV/  R(MK, MJ), U(MK, MJ), V(MK, MJ), P(MK, MJ), T(MK,
     1      MJ)
          COMMON  /FLUET/  F21(MK, 2), F22(MK, 2), F23(MK, 2), F25(MK, 2),
     1      G21(MK, 2), G22(MK, 2), G23(MK, 2), G25(MK, 2), H(MK)
          COMMON  /FLUZT/  F31(MK), F32(MK), F33(MK), F35(MK), G31(MK),
     1      G32(MK), G33(MK), G35(MK)
          COMMON  /NELSON/  RDET, RDZT, QDET, RMU, SL(MK, MJ)
          COMMON  /NOZ/  KNOZ, JNOZIN, JNOZOUT, JNHIN, JNHOUT, JNERIN,
     1      JNEROUT, PTNOZ, TTNOZ, ALPHA, EFFNOZ
          COMMON  /OLD/  ROLD(MK, MJ), RUOLD(MK, MJ), RVOLD(MK, MJ),
     1      REOLD(MK, MJ), XNORM
          COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
C
C         SPECIFY INITIAL CONDITIONS FOR THE FLOW FIELD
C
C
          K       = 1
          DO 1 J    = 1, JL
            R(K, J)   = RHO(K, J)
            U(K, J)   = RHOU(K, J) / RHO(K, J)
            V(K, J)   = RHOV(K, J) / RHO(K, J)
            T(K, J)   = (RHOE(K, J) / RHO(K, J) - 0.5 * (U(K, J) **
     1        2 + V(K, J) ** 2)) / CV
            P(K, J)   = R(K, J) * RC * T(K, J)
     1    CONTINUE
C
C         SPECIFY THE NO-SLIP BOUNDARY CONDITION AND SURFACE
C         TEMPERATURE
C
          DO 2 K    = 1, KL
            T(K, JLM)   = (RHOE(K, JLM) / RHO(K, JLM) - 0.5 * (U(K, JLM) **
     1        2 + V(K, JLM) ** 2)) / CV
            P(K, JLM)   = RHO(K, JLM) * RC * T(K, JLM)
            RHO(K, JL) = P(K, JLM) / (RC * TW)
            RHOU(K, JL) = 0.0
            RHOV(K, JL) = 0.0
            RHOE(K, JL) = RHO(K, JL) * CVTW
            R(K, JL)   = RHO(K, JL)
            U(K, JL)   = 0.0
            V(K, JL)   = 0.0
            T(K, JL)   = TW
            P(K, JL)   = P(K, JLM)
     2    CONTINUE
C
C         NOZZLE WALL NO-SLIP CONDITION AND TEMPERATURE
C
          DO 3 K    = 1, KNOZ
C
C           OUTSIDE OF INNER WALL
C
            T(K, JNERIN)    = (RHOE(K, JNERIN) / RHO(K, JNERIN) - 0.5 *
     1        (U(K,JNERIN) ** 2 + V(K, JNERIN) ** 2)) / CV
            P(K, JNERIN)    = RHO(K, JNERIN) * RC * T(K, JNERIN)
            RHO(K, JNHIN) = P(K, JNERIN) / (RC * TW)
            RHOU(K, JNHIN) = 0.0
            RHOV(K, JNHIN) = 0.0
            RHOE(K, JNHIN) = RHO(K, JNHIN) * CVTW
            R(K, JNHIN)   = RHO(K, JNHIN)
            U(K, JNHIN)   = 0.0
            V(K, JNHIN)   = 0.0
            T(K, JNHIN)   = TW
            P(K, JNHIN)   = P(K, JNERIN)
C
C           OUTSIDE OF OUTER WALL
C
            T(K, JNEROUT)    = (RHOE(K, JNEROUT) / RHO(K, JNEROUT) - 0.5 *
     1        (U(K,JNEROUT) ** 2 + V(K, JNEROUT) ** 2)) / CV
            P(K, JNEROUT)    = RHO(K, JNEROUT) * RC * T(K, JNEROUT)
            RHO(K, JNHOUT) = P(K, JNEROUT) / (RC * TW)
            RHOU(K, JNHOUT) = 0.0
            RHOV(K, JNHOUT) = 0.0
            RHOE(K, JNHOUT) = RHO(K, JNHOUT) * CVTW
            R(K, JNHOUT)   = RHO(K, JNHOUT)
            U(K, JNHOUT)   = 0.0
            V(K, JNHOUT)   = 0.0
            T(K, JNHOUT)   = TW
```

```
      P(K, JNHOUT)  = P(K, JNEROUT)
    3 CONTINUE
C
CC      INITIALIZE THE FLUX COMPONENTS
C
      DO 4 L    = 1, 2
        DO 4 K    = 1, KL
        F21(K, L) = 0.0
        F22(K, L) = 0.0
        F23(K, L) = 0.0
        F25(K, L) = 0.0
C
        G21(K, L) = 0.0
        G22(K, L) = 0.0
        G23(K, L) = 0.0
        G25(K, L) = 0.0
    4   CONTINUE
C
      DO 5 K    = 1, KL
      F31(K)    = 0.0
      F32(K)    = 0.0
      F33(K)    = 0.0
      F35(K)    = 0.0
C
      G31(K)    = 0.0
      G32(K)    = 0.0
      G33(K)    = 0.0
      G35(K)    = 0.0
      H(K)      = 0.0
    5 CONTINUE
      DO 6 J    = 1, JL
        DO 6 K    = 1, KL
        RHOP(K, J) = RHO(K, J)
        RHOUP(K, J) = RHOU(K, J)
        RHOVP(K, J) = RHOV(K, J)
        RHOEP(K, J) = RHOE(K, J)
        ROLD(K, J) = RHO(K, J)
        RUOLD(K, J) = RHOU(K, J)
        RVOLD(K, J) = RHOV(K, J)
        REOLD(K, J) = RHOE(K, J)
    6   CONTINUE
    XNORM     = 0.0
C
CC      GENERATE THE SCALING LENGTHS FOR USE IN EDDY.
C
      DO 7 J    = 1, JL
        DO 7 K    = 1, KL
        SL(K, J)  = SQRT((Y(K, J) - Y(K, JL)) ** 2 + (X(K, J) -
     1    X(K, JL)) ** 2)
    7   CONTINUE
      DO 8 J = 1, JNHIN
        DO 8 K = 1, KNOZ
        SL(K, J)  = SQRT((Y(K, J) - Y(K, JNHIN)) ** 2 +
     1    (X(K, J) - X(K, JNHIN)) ** 2)
    8 CONTINUE
      JCL = JNHOUT + (JL - JNHOUT) / 2
      DO 9 J = JNHOUT, JCL
        SL(K, J)  = SQRT((Y(K, J) - Y(K, JNHOUT)) ** 2 +
     1    (X(K, J) - X(K, JNHOUT)) ** 2)
    9 CONTINUE
      RETURN
```

```
      END
      SUBROUTINE  SUM (J)
      PARAMETER   (MK = 130, MJ = 115)
      COMMON // DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
     1  CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
     2  JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
      COMMON  /DAMP/  ADD1(MK), ADD2(MK), ADD3(MK), ADD5(MK)
      COMMON  /DEP/  RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
     1  MJ)
      COMMON  /DEPP/  RHOP(MK, MJ), RHOUP(MK, MJ), RHOVP(MK, MJ),
     1  RHOEP(MK, MJ)
      COMMON  /DOF/  X(MK, MJ), Y(MK, MJ), RY(MK, MJ)
      COMMON  /FLUET/  F21(MK, 2), F22(MK, 2), F23(MK, 2), F25(MK, 2),
     1  G21(MK, 2), G22(MK, 2), G23(MK, 2), G25(MK, 2), H(MK)
      COMMON  /FLUZT/  F31(MK), F32(MK), F33(MK), F35(MK), G31(MK),
     1  G32(MK), G33(MK), G35(MK)
      COMMON  /NONC/  G31N(2), G32N(2), G33N(2), G35N(2)
      COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
C
      IF (BETA .NE. 0.0) DT = DTL(J)
      RDET      = 1.0 / DETA
      RDZT      = 1.0 / DZETA
      KS        = KSTART(J)
      KSP       = KS + 1
C
      IF (L .EQ. 1) THEN
C
CC       PREDICTOR SWEEP
C
        DO 1 KV   = 2, KLM
        KM        = KV - 1
C
        RHOP(KV, J) = RHO(KV, J) - DT * ((ETX(KV, J) * (F21(KV, 2) -
     1    F21(KV, 1)) + ETY(KV, J) * RY(KV, J) * (G21(KV, 2) -
     2    G21(KV, 1))) * RDET + (ZTX(KV, J) * (F31(KV) - F31(KM)) +
     3    ZTY(KV, J) * RY(KV, J) * (G31(KV) - G31(KM))) * RDZT)
C
        RHOUP(KV, J) = RHOU(KV, J) - DT * ((ETX(KV, J) * (F22(KV, 2)
     1    - F22(KV, 1)) + ETY(KV, J) * RY(KV, J) * (G22(KV, 2) -
     2    G22(KV, 1))) * RDET + (ZTX(KV, J) * (F32(KV) - F32(KM)) +
     3    ZTY(KV, J) * RY(KV, J) * (G32(KV) - G32(KM))) * RDZT)
    1   CONTINUE
C
        ADDNC     =  - DT * ZTY(2, J) * (G31N(2) - G31N(1)) * RDZT
        RHOP(2, J) = RHOP(2, J) + ADDNC
        ADDNC     =  - DT * ZTY(2, J) * (G32N(2) - G32N(1)) * RDZT
        RHOUP(2, J) = RHOUP(2, J) + ADDNC
C
        DO 2 KV   = KSP, KLM
        KM        = KV - 1
C
        RHOVP(KV, J) = RHOV(KV, J) - DT * ((ETX(KV, J) * (F23(KV, 2)
     1    - F23(KV, 1)) + ETY(KV, J) * RY(KV, J) * (G23(KV, 2) -
     2    G23(KV, 1))) * RDET + (ZTX(KV, J) * (F33(KV) - F33(KM)) +
     3    ZTY(KV, J) * RY(KV, J) * (G33(KV) - G33(KM))) * RDZT +
     4    H(KV))
C
        RHOEP(KV, J) = RHOE(KV, J) - DT * ((ETX(KV, J) * (F25(KV, 2)
     1    - F25(KV, 1)) + ETY(KV, J) * RY(KV, J) * (G25(KV, 2) -
     2    G25(KV, 1))) * RDET + (ZTX(KV, J) * (F35(KV) - F35(KM)) +
     3    ZTY(KV, J) * RY(KV, J) * (G35(KV) - G35(KM))) * RDZT)
```

```
 2      CONTINUE
C
       ADDNC     =  - DT * ZTY(2, J) * (G33N(2) - G33N(1)) * RDZT
       RHOVP(2, J) = RHOVP(2, J) + ADDNC
       ADDNC     =  - DT * ZTY(2, J) * (G35N(2) - G35N(1)) * RDZT
       RHOEP(2, J) = RHOEP(2, J) + ADDNC
C
      IF ((ISMTHX .EQ. 0) .AND. (ISMTHY .EQ. 0)) RETURN
C
      CALL DAMPING (J)
C
       DO 3 KV   = KSP, KLM
       RHOP(KV, J) = RHOP(KV, J) + ADD1(KV)
       RHOUP(KV, J) = RHOUP(KV, J) + ADD2(KV)
       RHOVP(KV, J) = RHOVP(KV, J) + ADD3(KV)
       RHOEP(KV, J) = RHOEP(KV, J) + ADD5(KV)
 3     CONTINUE
C
    ELSE
C
C
C     CORRECTOR SWEEP
C
       DO 4 KV   = KSP, KLM
       KM        = KV - 1
C
       RHO(KV, J) = 0.5 * (RHO(KV, J) + RHOP(KV, J) - DT * ((ETX(KV,
     1   J) * (F21(KV, 2) - F21(KV, 1)) + ETY(KV, J) * RY(KV, J) *
     2   (G21(KV, 2) - G21(KV, 1))) * RDET + (ZTX(KV, J) * (F31(KV)
     3   - F31(KM)) + ZTY(KV, J) * RY(KV, J) * (G31(KV) - G31(KM)))
     4   * RDZT))
C
       RHOU(KV, J) = 0.5 * (RHOU(KV, J) + RHOUP(KV, J) - DT *
     1   ((ETX(KV, J) * (F22(KV, 2) - F22(KV, 1)) + ETY(KV, J) *
     2   RY(KV, J) * (G22(KV, 2) - G22(KV, 1))) * RDET + (ZTX(KV, J)
     3   * (F32(KV) - F32(KM)) + ZTY(KV, J) * RY(KV, J) * (G32(KV) -
     4   G32(KM))) * RDZT))
 4     CONTINUE
C
       DO 5 KV   = KSP, KLM
       KM        = KV - 1
C
       RHOV(KV, J) = 0.5 * (RHOV(KV, J) + RHOVP(KV, J) - DT *
     1   ((ETX(KV, J) * (F23(KV, 2) - F23(KV, 1)) + ETY(KV, J) *
     2   RY(KV, J) * (G23(KV, 2) - G23(KV, 1))) * RDET + (ZTX(KV, J)
     3   * (F33(KV) - F33(KM)) + ZTY(KV, J) * RY(KV, J) * (G33(KV) -
     4   G33(KM))) * RDZT + H(KV)))
C
       RHOE(KV, J) = 0.5 * (RHOE(KV, J) + RHOEP(KV, J) - DT *
     1   ((ETX(KV, J) * (F25(KV, 2) - F25(KV, 1)) + ETY(KV, J) *
     2   RY(KV, J) * (G25(KV, 2) - G25(KV, 1))) * RDET + (ZTX(KV, J)
     3   * (F35(KV) - F35(KM)) + ZTY(KV, J) * RY(KV, J) * (G35(KV) -
     4   G35(KM))) * RDZT))
 5     CONTINUE
C
      IF ((ISMTHX .EQ. 0) .AND. (ISMTHY .EQ. 0)) RETURN
C
      CALL DAMPING (J)
C
       DO 6 KV   = KSP, KLM
       RHO(KV, J) = RHO(KV, J) + ADD1(KV)
       RHOV(KV, J) = RHOV(KV, J) + ADD3(KV)
       RHOE(KV, J) = RHOE(KV, J) + ADD5(KV)
 6     CONTINUE
    END IF
    RETURN
    END
    SUBROUTINE  TMSTEP
    PARAMETER   (MK = 130, MJ = 115)
    COMMON // DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
   1  CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
   2  JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
    COMMON  /DEP/  RHO(MK, MJ), RHOU(MK, MJ), RHOV(MK, MJ), RHOE(MK,
   1  MJ)
    COMMON  /TV/  ETX(MK, MJ), ETY(MK  MJ), ZTX(MK, MJ), ZTY(MK, MJ)
    COMMON  /TVIS/  EP(MK, MJ), CPQPR, CPQPRT
C
C       SET UP INITIAL CFL TIME STEP VALUE
C
    DIMENSION   C(MK), DTC(MK), U(MK), UET(MK), UZT(MK), V(MK)
C
    DTC(1)    = 1.0
    GAMM2     = GAMMA * GAMM1
    DTCFL     = 1.0
    RDET      = 1.0 / DETA
    RDZT      = 1.0 / DZETA
    CONST     = SQRT(2.0 / 3.0)
    RPRT      = 1.0 / PRT
    RPR       = 1.0 / PR
    THOGAM    = 2.0 * GAMMA
    RCG       = RC * GAMMA
C
      DO 3 J    = 2, JLM
        KSP     = KSTART(J) + 1
        DO 1 K    = KSP, KLM
        RH        = 1.0 / RHO(K, J)
        U(K)      = RHOU(K, J) * RH
        V(K)      = RHOV(K, J) * RH
        C(K)      = SQRT(GAMM2 * (RHOE(K, J) * RH - 0.5 * (U(K) ** 2
     1    + V(K) ** 2)))
        TT        = C(K) * C(K) / RCG
        RMU       = 2.27E - 8 * SQRT(TT ** 3) / (TT + 19^ 6)
C
        UET(K)    = ETX(K, J) * U(K) + ETY(K, J) * V(K)
        UZT(K)    = ZTX(K, J) * U(K) + ZTY(K, J) * V(K)
C
        DT1       = 1.0 / (ABS(UET(K)) * RDET + ABS(UZT(K)) * RDZT +
     1    C(K) * SQRT((ETX(K, J) * RDET + ZTX(K, J) * RDZT) ** 2 +
     2    (ETY(K, J) * RDET + ZTY(K, J) * RDZT) ** 2))
C
        DE        = RDET * SQRT(ETX(K, J) ** 2 + ETY(K, J) ** 2)
        DZ        = R ZT * SQRT(ZTX(K, J) ** 2 + ZTY(K, J) ** 2)
        DT2       = 1.0 / (ABS(UET(K)) * RDET + C(K) * DE + RH * DE *
     1    (THOGAM * DE * (RMU * RPR + EP(K, J) * RPRT) + CONST * DZ *
     2    (RMU + EP(K, J))))
C
        DT3       = 1.0 / (ABS(UZT(K)) * RDZT + C(K) * DZ + RH * DZ *
     1    (THOGAM * DZ * (RMU * RPR + EP(K, J) * RPRT) + CONST * DE *
     2    (RMU + EP(K, J))))
        DTC(K)    = MIN(DT1, DT2, DT3)
 1      CONTINUE
C
```

```
C
C         PERFORM THE COMPARISON AND MODIFY THE CFL TIME STEP
C
          DTMIN     = DTC(1)
            DO 2 K    = KSP, KLM
            DTMIN     = MIN(DTC(K), DTMIN)
      2     CONTINUE
C
          DTL(J)    = DTMIN * CFL
C
C
C         COMPARE DTMIN BETWEEN ADJACENT PLANES
C
          DTCFL     = MIN(DTCFL, DTMIN)
      3   CONTINUE
C
C
C         ADJUST DTCFL FOR VISCOUS EFFECT (TRIAL AND ERROR)
C
        DT        = CFL * DTCFL
        RETURN
        END
        SUBROUTINE  TRANS
        PARAMETER   (MK = 130, MJ = 115)
        COMMON // DETA, DZETA, CV, RC, PR, PRT, GAMMA, DTL(MJ), GAMM1,
       1  CFL, BETA, UINF, RHOINF, CINF, TINF, TH, DT, CX, CY, L, JL, KL,
       2  JLM, KLM, ISMTHX, ISMTHY, KSTART(MJ)
        COMMON  /DOF/  X(MK, MJ), Y(MK, MJ), RY(MK, MJ)
        COMMON  /NOZ/  KNOZ, JNOZIN, JNOZOUT, JNWIN, JNWOUT, JNERIN,
       1  JNEROUT, PTNOZ, TTNOZ, ALPHA, EFFNOZ
        COMMON  /TV/  ETX(MK, MJ), ETY(MK, MJ), ZTX(MK, MJ), ZTY(MK, MJ)
C
        DIMENSION   DJ(MK, MJ), RDJ(MK, MJ), XET(MK, MJ), XZT(MK, MJ),
       1  YET(MK, MJ), YZT(MK, MJ)
C
        RDET      = 1.0 / (2.0 * DETA)
        RDZT      = 1.0 / (2.0 * DZETA)
        JLM2      = JLM - 1
        KLM2      = KLM - 1
C
C
C         GENERATE DX/DETA AND DY/DETA
C
C         ONE-SIDED DIFFERENCING FOR J = 1
C
          DO 1 KV   = 1, KL
          XET(KV, 1) = (4.0 * X(KV, 2) - X(KV, 3) - 3.0 * X(KV, 1)) *
       1    RDET
          YET(KV, 1) = (4.0 * Y(KV, 2) - Y(KV, 3) - 3.0 * Y(KV, 1)) *
       1    RDET
      1   CONTINUE
C
C         ONE-SIDED DIFFERENCING FOR J = JL
C
          DO 2 KV   = 1, KL
          XET(KV, JL) =  - (4.0 * X(KV, JLM) - X(KV, JLM2) - 3.0 * X(KV,
       1    JL)) * RDET
          YET(KV, JL) =  - (4.0 * Y(KV, JLM) - Y(KV, JLM2) - 3.0 * Y(KV,
       1    JL)) * RDET
      2   CONTINUE
C
C         CENTRAL DIFFERENCING FOR FIELD POINTS
C
          DO 4 JV   = 2, JLM
          JP        = JV + 1
          JM        = JV - 1
          KS        = 1
            DO 3 KV   = KS, KL
            XET(KV, JV) = (X(KV, JP) - X(KV, JM)) * RDET
            YET(KV, JV) = (Y(KV, JP) - Y(KV, JM)) * RDET
      3     CONTINUE
      4   CONTINUE
C
C         ONE-SIDED DIFFERENCING FOR J = JNWOUT
C
          JV = JNWOUT
          JVP = JV + 1
          JVP2 = JV + 2
          DO 5 KV   = 1, KNOZ
          XET(KV, JV) = (4.0 * X(KV, JVP) - X(KV, JVP2) - 3.0 *
       1    X(KV, JV)) * RDET
          YET(KV, JV) = (4.0 * Y(KV, JVP) - Y(KV, JVP2) - 3.0 *
       1    Y(KV, JV)) * RDET
      5   CONTINUE
C
C         ONE-SIDED DIFFERENCING FOR J = JNWIN
C
          JV = JNWIN
          JVM = JV - 1
          JVM2 = JV - 2
          DO 6 KV   = 1, KNOZ
          XET(KV, JV) =  - (4.0 * X(KV, JVM) - X(KV, JVM2) - 3.0 * X(KV,
       1    JV)) * RDET
          YET(KV, JV) =  - (4.0 * Y(KV, JVM) - Y(KV, JVM2) - 3.0 * Y(KV,
       1    JV)) * RDET
      6   CONTINUE
C
C         GENERATE DX/DZETA AND DY/DZETA
C
C         ONE-SIDED DIFFERENCING FOR K = 1
C
          DO 7 JV   = 1, JL
          XZT(1, JV) = (4.0 * X(2, JV) - X(3, JV) - 3.0 *
       1    X(1, JV)) * RDZT
          YZT(1, JV) = (4.0 * Y(2, JV) - Y(3, JV) - 3.0 *
       1    Y(1, JV)) * RDZT
      7   CONTINUE
C
C         ONE-SIDED DIFFERENCING FOR K = KL
C
          DO 8 JV   = 1, JL
          XZT(KL, JV) =  - (4.0 * X(KLM, JV) - X(KLM2, JV) - 3.0 * X(KL,
       1    JV)) * RDZT
          YZT(KL, JV) =  - (4.0 * Y(KLM, JV) - Y(KLM2, JV) - 3.0 * Y(KL,
       1    JV)) * RDZT
      8   CONTINUE
C
C         CENTRAL DIFFERENCING FOR FIELD POINTS
C
          DO 10 JV   = 1, JL
            DO 9 KV   = 2, KLM
            KP        = KV + 1
            KM        = KV - 1
            XZT(KV, JV) = (X(KP, JV) - X(KM, JV)) * RDZT
            YZT(KV, JV) = (Y(KP, JV) - Y(KM, JV)) * RDZT
      9     CONTINUE
```

```
   10    CONTINUE
C
C         ONE-SIDED DIFFERENCING FOR K = KNOZ
C
         DO 11 JV   = JNMIN, JNMOUT
         KS   = KNOZ
         KSP  = KS + 1
         KSP2 = KS + 2
         XZT(KS, JV) = (4.0 * X(KSP, JV) - X(KSP2, JV) - 3.0 *
     1     X(KS, JV)) * RDZT
         YZT(KS, JV) = (4.0 * Y(KSP, JV) - Y(KSP2, JV) - 3.0 *
     1     Y(KS, JV)) * RDZT
   11    CONTINUE
C
C         GENERATE THE INVERSE JACOBIAN OF TRANSFORMATION
C
         DO 13 JV  = 1, JL
           DO 12 KV  = 1, KL
           DJ(KV, JV) = XET(KV, JV) * YZT(KV, JV) - XZT(KV, JV) *
     1       YET(KV, JV)
   12      CONTINUE
   13    CONTINUE
C
         DO 15 JV  = 1, JL
           DO 14 KV  = 1, KL
           RDJ(KV, JV) = 1.0 / DJ(KV, JV)
   14      CONTINUE
   15    CONTINUE
C
C         GENERATE THE METRICS OF THE COORDINATE TRANSFORMATION
C
         DO 17 JV  = 1, JL
           DO 16 KV  = 1, KL
           ETX(KV, JV) = YZT(KV, JV) * RDJ(KV, JV)
           ZTX(KV, JV) =  - YET(KV, JV) * RDJ(KV, JV)
           ETY(KV, JV) =  - XZT(KV, JV) * RDJ(KV, JV)
           ZTY(KV, JV) = XET(KV, JV) * RDJ(KV, JV)
   16      CONTINUE
   17    CONTINUE
C
C         GENERATE THE 1 / R TERM
C
         DO 18 KV  = 1, KL
         RY(KV, 1) = 0.0
           DO 18 JV  = 2, JL
           RY(KV, JV) = 1.0 / Y(KV, JV)
   18      CONTINUE
       RETURN
       END
```

Appendix C: Post-Processing Program *AUGMENT*

```
      PROGRAM    AUGMENT
      PARAMETER  (JL = 115, KL = 130, KNOZ =46)
C
C       COMPUTES THE THRUST AUGMENTATION BY INTEGRATING
C       THE X-MOMENTUM AT THE EXIT COMPARING THE
C       RESULTING THRUST TO THE ISENTROPIC THRUST.
C
      COMMON  /RAWDATA/  RJ(JL), U(JL), RHO(JL), RHOREF, UREF
C
C       WRITE A HEADER
C
      OPEN (6, FILE = 'EJTHRST', FORM = 'FORMATTED', STATUS = 'NEW')
      WRITE(6,100)
      WRITE(6,110)
      WRITE(6,120)
      WRITE(6,130)
      WRITE(6,140)
C
      JLM = JL - 1
C
      VELNOZ = 478.725
      RHONOZ = 2.465687 E-03
      AREANOZ = 0.9066386 E-02
C
C       READ DATA
C
      CALL READDAT
C
C       CALCULATE X-MOMENTUM
C
      R = 0.5 * RJ(2)
      XMOM = 3.14159 * R * R * RHO(1) * U(1) * U(1)
      DO 10 J = 2, JLM
        JP = J + 1
        JM = J - 1
        DELTAR = 0.5 * (RJ(JP) - RJ(JM))
        XMOM = XMOM + 6.283185 * RJ(J) * DELTAR * RHO(J) *
     1    U(J) * U(J)
   10 CONTINUE
C
C       CALCULATE THRUST AND REFERENCE (ISENTROPIC) THRUST
C       AND GET THRUST AUGMENTATION RATIO
C
      THRST = XMOM
      RFTHRST = RHONOZ * VELNOZ * VELNOZ * AREANOZ
      PHI = THRST / RFTHRST
C
C       OUTPUT THRUST, REFERENCE THRUST AND THRUST
C       AUGMENTATION RATIO
C
      WRITE(6,150)
      WRITE(6,140)
      WRITE(6,160) THRST, RFTHRST, PHI
      STOP
C
  100 FORMAT ( ' THIS PROGRAM CALCULATES THE THRUST GENERATED' )
  110 FORMAT ( ' BY THE EJECTOR AND COMPARES IT TO THE ISENTROPIC' )
  120 FORMAT ( ' THRUST OF THE NOZZLE ALONE TO ARRIVE AT A THRUST' )
  130 FORMAT ( ' AUGMENTATION RATIO.' )
  140 FORMAT ( ' ' )
  150 FORMAT ( 4X, 'MOM THRUST', 4X, 'ISEN THRUST', 6X, 'PHI' )
  160 FORMAT ( 3F13.5 )
C
      END
      SUBROUTINE  READDAT
      PARAMETER  (JL = 115, KL = 130, KNOZ =46)
      COMMON  /RAWDATA/  RJ(JL), U(JL), RHO(JL), RHOREF, UREF

      OPEN (64, FILE = 'EJTEMP', FORM = 'FORMATTED', STATUS = 'OLD')
      READ (64,200)
      READ (64,200) RHOREF
      READ (64,210) UREF
      READ (64,200)
      READ (64,200)
C
C         READ DATA FOR EXIT PLANE
C
      DO 10 J = 1, JL
        READ (64,220) RJ(J), RHO(J), U(J)
        RHO(J) = RHO(J) * RHOREF
        U(J) = U(J) * UREF
   10 CONTINUE
C
  200 FORMAT ( 31X, E15.7 )
  210 FORMAT ( 31X, F10.4 )
  220 FORMAT ( 18X, G11.5, G13.5, G11.5
C
      END
```

Bibliography

1. Anderson, D. A., J. C. Tannehill and R. H. Pletcher. Computational Fluid Mechanics and Heat Transfer. New York: Hemisphere Publishing Corporation, 1984.

2. Baldwin, B. S. and H. Lomax. 'Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows,' AIAA Paper 78-257 (January 1978).

3. Hankey, W. L. and J. S. Shang. 'Natural Transition - A Self Excited Oscillation,' AIAA Paper 82-1011 (June 1982).

4. Hasen, Capt. Gerald A. 'Navier-Stokes Solutions for an Axisymmetric Nozzle in a Supersonic External Stream,' AFWAL Technical Report 81-3161 (March 1982).

5. Hill, P. G. and C. R. Peterson. Mechanics and Thermodynamics of Propulsion. Reading, Massachusetts: Addison-Wesley Publishing Company, 1965.

6. Kedem, Maj. Eli. Israeli Air Force. An Experimental Study of Static Thrust Augmentation Using a 2-D Variable Ejector. MS Thesis AFIT/GAE/AA/79D-7. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1979.

7. Kohlman, D. L. Introduction to V/STOL Airplanes. Ames, Iowa: Iowa State University Press, 1981.

8. Lewis, Capt. William D. US Army. An Experimental Study of Thrust Augmenting Ejectors. MS Thesis AFIT/GAE/AA/83D-13. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1983.

Bibliography (cont.)

9. McCormick, B. W. Jr. Aerodynamics of V/STOL Flight. Orlando, Florida: Academic Press, 1967.

10. Olson, L. E., P. R. McGowan and R. W. MacCormack. 'Numerical Solution of the Time-Dependent Compressible Navier-Stokes Equations in Inlet Regions,' NASA TM-X-62338 (March 1974)

11. Reznick, Capt. Steven G. An Experimental Study of Circular and Rectangular Thrust Augmenting Ejectors. MS Thesis AFIT/GAE/AA/80D-18. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1980.

12. Roache, P. J. Computational Fluid Dynamics. Albuquerque, New Mexico: Hermosa Publishers, 1972.

13. Shang, J. S., W. L. Hankey, and R. E. Smith. 'Flow Oscillations of Spike-Tipped Bodies,' AIAA Paper 80-0062 (January 1980).

14. Shang, J. S. and R. W. MacCormack. 'Flow Over a Biconic Configuration with an Afterbody Compression Flap,' AFWAL Technical Report 84-3059 (April 1984).

15. Stock, H. W. and W. Haase. 'Determination of Length Scales in Algebraic Turbulence Models for Navier-Stokes Methods,' AIAA Journal Vol 27, No 1: 5-14 (January 1989).

Bibliography (cont.)

16. Uhuad, Capt. Generoso C. An Experimental Investigation of a Circular Thrust Augmenting Ejector. MS Thesis AFIT/GAE/AA/86M-3. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1985.

17. Unnever, Capt. Gregory. An Experimental Study of Rectangular and Circular Thrust Augmenting Ejectors. MS Thesis AFIT/GAE/AA/81D-31. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1981.

18. Von Kármán, T. "Theoretical Remarks on Thrust Augmentation," Contributions to Applied Mechanics, Reissner Anniversary Volume, Ann Arbor, Michigan, pp 461-468, 1949.

19. Zucrow, M. J. and J. D. Hoffman. Gas Dynamics Vol 1, New York: John Wiley & Sons, 1976.

Vita

Captain Kenneth R. Gage [redacted] He moved to Branford, Connecticut in 1965 and graduated from Branford High School in 1980. He was accepted to and attended the United States Air Force Academy in Colorado Springs, Colorado, from which he received the degree of Bachelor of Science in Astronautical Engineering and a regular commission in the USAF on 30 May 1984. He then went to Headquarters, USAF Space Division (AFSC) at Los Angeles AFB where he served as a flight operations manager for the Inertial Upper Stage and as an advanced launch systems project officer until entering the School of Engineering, Air Force Institute of Technology, in May 1988.

[redacted]

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1990	Master's Thesis

4. TITLE AND SUBTITLE

Numerical Analysis of an Axisymmetric Thrust Augmenting Ejector

5. FUNDING NUMBERS

6. AUTHOR(S)

Kenneth R. Gage, Capt, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GAE/ENY/89D-10

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Flight Dynamics Laboratory
WRDC/FIMM
Wright Research and Development Center
Wright-Patterson AFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release;
distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Use of an ejector is an effective way to increase the thrust produced by a jet. In this thesis project an axisymmetric ejector concept which has been previously explored by experiment was numerically modeled. An existing axisymmetric, internal flow code based on the explicit MacCormack method was modified to incorporate primary nozzle structure and flow injection within the flowfield. Results were compared qualitatively and quantitatively with experimental results to verify the validity of the model. Internal flow structure, difficult to obtain in experiment, is easily examined. This code may be used for parametric analysis of such ejector performance parameters as primary nozzle location, flow injection angle, diffuser area ratio, nozzle area ratio, and inlet geometry to optimize future hardware configurations.

14. SUBJECT TERMS			15. NUMBER OF PAGES
Fluid Dynamics Fluid Flow	Axially Symmetric Flow Jet Mixing Flow	Air Ejectors Computational Fluid Dynamics (CFD)	104 16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to **stay within the lines to meet optical scanning requirements.**

Block 1. Agency Use Only (Leave Blank)

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	**PR**	- Project
G	- Grant	**TA**	- Task
PE	- Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Names(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency. Report Number. (If known)

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of ..., To be published in .... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availablity Statement. Denote public availability or limitation. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR)

DOD - See DoDD 5230.24, "Distribution Statements on Technical Documents."
DOE - See authorities
NASA - See Handbook NHB 2200.2.
NTIS - Leave blank.

Block 12b. Distribution Code.

DOD - DOD - Leave blank
DOE - DOE - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports
NASA - NASA - Leave blank
NTIS - NTIS - Leave blank.

Block 13. Abstract. Include a brief (Maximum 200 words) factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code (NTIS only).

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.